Oracle® HTTP Server

Administration Guide

Release 2 (9.0.2)

May 2002

Part No. A92173-02

Oracle HTTP Server Administration Guide, Release 2 (9.0.2)

Part No. A92173-02

Primary Authors: Priya Darshane, Julia Pond

Contributors: Warren Briese, Ling Cheng, Gary Hallmark, Bob Hanckel, John Janosik, Sharon Lee, Dan Mullen, Chuck Murray, Mark Nelson, Zhiyin Pan, Udayini Pendyala, Mike Rubino, Shirley Ann Stern, Liz Trojan, Huiping Wang, Kevin Wang

Contents

5 Configuring and Using Server Logs

6 Oracle HTTP Server Modules

7 Configuring and Using mod_oradav

B Third Party Licenses

Index

Send Us Your Comments

Oracle HTTP Server Administration Guide, Release 2 (9.0.2)

Part No. A92173-02

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: iasdocs_us@oracle.com
- FAX: 650-506-7407 Attn: Oracle9*i* Application Server Documentation Manager
- Postal service:

  Oracle Corporation
  Oracle9*i* Application Server Documentation
  500 Oracle Parkway, M/S 2op3
  Redwood Shores, CA 94065
  USA

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

This guide describes how to administer the Oracle HTTP Server.

This preface contains these topics:

- Audience
- Documentation Accessibility
- Organization
- Related Documentation
- Conventions

Audience

The *Oracle HTTP Server Administration Guide* is intended for application server administrators, security managers, and managers of databases used by application servers.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle Corporation is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at

`http://www.oracle.com/accessibility/`

Accessibility of Links to External Web Sites in Documentation This documentation may contain links to Web sites of other companies or organizations that Oracle Corporation does not own or control. Oracle Corporation neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Organization

This document contains:

Chapter 1, "Overview"

This chapter provides an overview of the Oracle HTTP Server, highlights the differences between the Oracle distribution and the open source Apache product, and explain how to start, stop, restart the server and access the Oracle HTTP Server default page.

Chapter 2, "Specifying the Server and File Locations"

This chapter explain how to set the server and server administrator options, and specifies various file locations.

Chapter 3, "Managing Server Processes"

This chapter provides an overview of the Oracle HTTP Server processes and provides information on how to regulate, and monitor these process.

Chapter 4, "Managing the Network Connection"

This chapter provides information about specifying IP addresses and ports, and managing server interaction and network connection persistence.

Chapter 5, "Configuring and Using Server Logs"

This chapter discusses the log formats for the Oracle HTTP Server, and describes the various log files and their locations.

Chapter 6, "Oracle HTTP Server Modules"

This chapter describes the modules (mods) included in the Oracle HTTP Server

Chapter 7, "Configuring and Using mod_oradav"

This chapter provides information to help you configure and use `mod_oradav`, so that you can use `OraDAV` to access content in an Oracle database from a Web browser or a `WebDAV` client.

Chapter 8, "Frequently Asked Questions"

This chapter provides answers to frequently asked questions on how to configure the Oracle HTTP Server to perform specialized useful functions.

Chapter A, "Using the Oracle9iAS Proxy Plug-in"

This appendix explains how the Oracle9*i*AS Proxy Plug-in enables you to use Oracle9*i*AS components in conjunction with a third-party HTTP listener.

Chapter B, "Third Party Licenses"

This appendix includes the Third Party License for all the third party products included with Oracle9*i* Application Server.

Related Documentation

For more information, see these Oracle resources:

- Oracle9*i* Application Server Documentation Library

- Oracle9*i* Application Server Platform-Specific Documentation on Oracle9*i* Application Server Disk 1

In North America, printed documentation is available for sale in the Oracle Store at

`http://oraclestore.oracle.com/`

Customers in Europe, the Middle East, and Africa (EMEA) can purchase documentation from

`http://www.oraclebookshop.com/`

Other customers can contact their Oracle representative to purchase printed documentation.

To download free release notes, installation documentation, white papers, or other collateral, please visit the Oracle Technology Network (OTN). You must register online before using OTN; registration is free and can be done at

`http://otn.oracle.com/admin/account/membership.html`

If you already have a username and password for OTN, then you can go directly to the documentation section of the OTN Web site at

`http://otn.oracle.com/docs/index.htm`

Conventions

This section describes the conventions used in the text and code examples of this documentation set. It describes:

- Conventions in Text
- Conventions in Code Examples
- Conventions for Microsoft Windows Operating Systems

Conventions in Text

We use various conventions in text to help you more quickly identify special terms. The following table describes those conventions and provides examples of their use.

Convention	Meaning	Example
Bold	Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.	When you specify this clause, you create an **index-organized table**.
Italics	Italic typeface indicates book titles or emphasis.	*Oracle9i Database Concepts* Ensure that the recovery catalog and target database do *not* reside on the same disk.
`UPPERCASE monospace (fixed-width) font`	Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, RMAN keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, usernames, and roles.	You can specify this clause only for a `NUMBER` column. You can back up the database by using the `BACKUP` command. Query the `TABLE_NAME` column in the `USER_TABLES` data dictionary view. Use the `DBMS_STATS.GENERATE_STATS` procedure.
`lowercase monospace (fixed-width) font`	Lowercase monospace typeface indicates executables, filenames, directory names, and sample user-supplied elements. Such elements include computer and database names, net service names, and connect identifiers, as well as user-supplied database objects and structures, column names, packages and classes, usernames and roles, program units, and parameter values. **Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	Enter `sqlplus` to open SQL*Plus. The password is specified in the `orapwd` file. Back up the datafiles and control files in the `/disk1/oracle/dbs` directory. The `department_id`, `department_name`, and `location_id` columns are in the `hr.departments` table. Set the `QUERY_REWRITE_ENABLED` initialization parameter to `true.` Connect as `oe` user. The `JRepUtil` class implements these methods.
`lowercase italic monospace (fixed-width) font`	Lowercase italic monospace font represents placeholders or variables.	You can specify the *`parallel_clause`*. Run `U`*`old_release`*`.SQL` where *`old_release`* refers to the release you installed prior to upgrading.

Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

Convention	Meaning	Example
[ ]	Brackets enclose one or more optional items. Do not enter the brackets.	DECIMAL (*digits* [, *precision*])
{ }	Braces enclose two or more items, one of which is required. Do not enter the braces.	{ENABLE | DISABLE}
|	A vertical bar represents a choice of two or more options within brackets or braces. Enter one of the options. Do not enter the vertical bar.	{ENABLE | DISABLE} [COMPRESS | NOCOMPRESS]
. . .	Horizontal ellipsis points indicate either:	
	■ That we have omitted parts of the code that are not directly related to the example	CREATE TABLE ... AS *subquery*;
	■ That you can repeat a portion of the code	SELECT *col1*, *col2*, ... , *coln* FROM employees;
. . .	Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example.	

Other notation	You must enter symbols other than brackets, braces, vertical bars, and ellipsis points as shown.	acctbal NUMBER(11,2);
		acct CONSTANT NUMBER(4) := 3;

Italics	Italicized text indicates placeholders or variables for which you must supply particular values.	CONNECT SYSTEM/*system_password* DB_NAME = *database_name*

Convention	Meaning	Example

`UPPERCASE`	Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. However, because these terms are not case sensitive, you can enter them in lowercase.	`SELECT last_name, employee_id FROM employees;`
		`SELECT * FROM USER_TABLES;`
		`DROP TABLE hr.employees;`

`lowercase`	Lowercase typeface indicates programmatic elements that you supply. For example, lowercase indicates names of tables, columns, or files.
	Note: Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.
		`SELECT last_name, employee_id FROM employees;`
		`sqlplus hr/hr`
		`CREATE USER mjones IDENTIFIED BY ty3MU9;`

Conventions for Microsoft Windows Operating Systems

The following table describes conventions for Microsoft Windows operating systems and provides examples of their use.

Convention	Meaning	Example
Choose Start >	How to start a program.	To start the Oracle Database Configuration Assistant, choose Start > Programs > Oracle - *HOME_NAME* > Configuration and Migration Tools > Database Configuration Assistant.
File and directory names	File and directory names are not case sensitive. The following special characters are not allowed: left angle bracket (<), right angle bracket (>), colon (:), double quotation marks ("), slash (/), pipe (|), and dash (-). The special character backslash (\) is treated as an element separator, even when it appears in quotes. If the file name begins with \\, then Windows assumes it uses the Universal Naming Convention.	`c:\winnt"\"system32` is the same as `C:\WINNT\SYSTEM32`

Convention	Meaning	Example
`C:\>`	Represents the Windows command prompt of the current hard disk drive. The escape character in a command prompt is the caret (^). Your prompt reflects the subdirectory in which you are working. Referred to as the *command prompt* in this manual.	`C:\oracle\oradata>`

	The backslash (\) special character is sometimes required as an escape character for the double quotation mark (") special character at the Windows command prompt. Parentheses and the single quotation mark (') do not require an escape character. Refer to your Windows operating system documentation for more information on escape and special characters.	`C:\>exp scott/tiger TABLES=emp QUERY=\"WHERE job='SALESMAN' and sal<1600\"`
		`C:\>imp SYSTEM/`*`password`*` FROMUSER=scott TABLES=(emp, dept)`

`HOME_NAME`	Represents the Oracle home name. The home name can be up to 16 alphanumeric characters. The only special character allowed in the home name is the underscore.	`C:\> net start Oracle`*`HOME_NAME`*`TNSListener`

Convention	Meaning	Example
ORACLE_HOME and *ORACLE_BASE*	In releases prior to Oracle8*i* release 8.1.3, when you installed Oracle components, all subdirectories were located under a top level *ORACLE_HOME* directory that by default used one of the following names: ■ `C:\orant` for Windows NT ■ `C:\orawin95` for Windows 95 ■ `C:\orawin98` for Windows 98 This release complies with Optimal Flexible Architecture (OFA) guidelines. All subdirectories are not under a top level *ORACLE_HOME* directory. There is a top level directory called *ORACLE_BASE* that by default is `C:\oracle`. If you install Oracle9*i* release 1 (9.0.1) on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is `C:\oracle\ora90`. The Oracle home directory is located directly under *ORACLE_BASE*. All directory path examples in this guide follow OFA conventions. Refer to *Oracle9i Database Getting Starting for Windows* for additional information about OFA compliances and for information about installing Oracle products in non-OFA compliant directories.	Go to the *ORACLE_BASE*`\`*ORACLE_HOME*`\rdbms\admin` directory.

1 Overview

This chapter describes the Oracle HTTP Server, highlighting the differences between the Oracle distribution and the open source Apache product on which it is based. It also explains how to start the server, access the Oracle HTTP Server main page, and stop and restart the server.

> **Note:** You can use Oracle Enterprise Manager for administering the Oracle HTTP Server. Oracle Enterprise Manager provides a Web-based tool that allows you to perform some of the management tasks described in this book. For more information, see the *Oracle9i Application Server Administrator's Guide*.

This chapter contains the following topics:

- Oracle HTTP Server Features
- Oracle HTTP Server Support
- Starting, Stopping, and Restarting the Oracle HTTP Server
- Starting the Oracle HTTP Server
- Stopping the Oracle HTTP Server
- Restarting the Oracle HTTP Server

Oracle HTTP Server Features

The Oracle HTTP Server provides a robust, reliable web server (that is based on the Apache HTTP Server version 1.3.22), pre-configured to:

- Provide a Servlet 2.3 container with Oracle9*i*AS Containers for J2EE (OC4J).
- Provide a high availability infrastructure, Oracle Process Management and Notification (OPMN), for process management, death detection and failover for OC4J and Oracle HTTP Server processes.
- Provide single sign-on capability.

  See Also: *Oracle9i Application Server Security Guide*.

- Enable securing of transactions with Secure Sockets Layer (SSL) technology.

  See Also: *Oracle9i Application Server Security Guide*.

- Execute Perl scripts in the same process as the Oracle HTTP Server.
- Access database stored procedures with a PL/SQL engine.

  See Also: *Oracle9i Application Server mod_plsql Users Guide*

- Enable scripting of HTML pages with PL/SQL code.
- Provide infrastructure for Business Components for Java (BC4J).
- Support the Java Server Pages (JSP) specification v.1.1.
- Support legacy use of Apache JServ, including a process management and death detection module (mod_oprocmgr).

Oracle HTTP Server Support

Oracle provides technical support for the following HTTP Server features and conditions:

- Modules included in the Oracle distribution, except as noted in the table in "Oracle HTTP Server Modules". Modules from any other source, including the Apache Software Foundation, are not supported by Oracle.
- Problems that can be reproduced within an Apache configuration consisting only of supported Oracle Apache modules.

- Use of the included Perl interpreter within the supported Apache configuration only.

Oracle HTTP Server Modules

The table below identifies the modules shipped with the Oracle HTTP Server. Note that the list differs from the Apache open source distribution (given the inclusion of Oracle modules), and that not all modules are supported by Oracle.

Table 1–1 Oracle HTTP Server Modules

Module	Oracle Support	Notes
mod_access	Yes	UNIX systems only.
mod_actions	Yes	
mod_alias	Yes	
mod_asis	No	
mod_auth	Yes	
mod_auth_anon	Yes	
mod_auth_db	No	Disabled. Not shipped by Oracle.
mod_auth_dbm	No	
mod_auth_digest	No	Disabled. Experimental MD5 authentication; not shipped by Oracle.
mod_autoindex	Yes	
mod_cern_meta	No	
mod_cgi	Yes	
mod_define	Yes	UNIX systems only.
mod_digest	Yes	
mod_dir	Yes	
mod_dms	Yes	Oracle module.
mod_env	Yes	
mod_example	No	
mod_expires	Yes	

Table 1–1 Oracle HTTP Server Modules (Cont.)

Module	Oracle Support	Notes
mod_fastcgi	**Yes**	
mod_headers	**Yes**	
mod_imap	**No**	
mod_include	**Yes**	
mod_info	**Yes**	
mod_isapi	**No**	
mod_jserv	**Yes**	Disabled by default in Oracle configuration.
mod_log_agent	**No**	Deprecated.
mod_log_config	**Yes**	
mod_log_referer	**Yes**	Deprecated.
mod_mime	**Yes**	
mod_mime_magic	**Yes**	
mod_mmap_static	**No**	Not shipped by Oracle.
mod_negotiation	**Yes**	
mod_oc4j	**Yes**	Oracle module. Recommended servlet container; enabled by default in Oracle configuration.
mod_oprocmgr	**Yes**	Oracle module.
mod_oradav	**Yes**	Oracle module.
mod_ossl	**Yes**	Oracle module.
mod_osso	**Yes**	Oracle module.
mod_perl	**Yes**	Third-party module.
mod_plsql	**Yes**	Oracle module.
mod_proxy	**Yes**	
mod_rewrite	**Yes**	
mod_setenvif	**Yes**	
mod_so	**Yes**	
mod_speling	**Yes**	

Table 1–1 Oracle HTTP Server Modules (Cont.)

Module	Oracle Support	Notes
mod_status	**Yes**	
mod_unique_id	**Yes**	UNIX systems only.
mod_userdir	**Yes**	
mod_usertrack	**Yes**	
mod_vhost_alias	**Yes**	

Starting, Stopping, and Restarting the Oracle HTTP Server

Oracle HTTP Server is managed by Distributed Configuration Management (DCM). There are two ways to access DCM: through the Oracle Enterprise Manager graphical user interface, and the command-line utility dcmctl, located in `ORACLE_HOME/dcm/bin` (UNIX) or `ORACLE_HOME\dcm\bin` (Windows).

See Also: *Oracle9i Application Server Administrator's Guide*

You must always use DCM to start, stop and restart the Oracle HTTP Server. Otherwise, the configuration management infrastructure cannot detect or communicate with the Oracle HTTP Server processes, and problems may occur. Do not use the `apachectl` utility to manage the Oracle HTTP Server.

To determine the state of the Oracle HTTP Server, use the `getstate` command with the verbose option:

```
dcmctl getstate -v
```

The processes are listed with their current state (Up, Down, etc.)

The dcmctl commands are listed in Table 1–2.

Table 1–2 dcmctl commands

Command	Result
`dcmctl start -ct ohs`	Starts the Oracle HTTP Server process in the local instance.
`dcmctl restart -ct ohs`	Restarts the Oracle HTTP Server process in the local instance (’graceful’ restart).
`dcmctl stop -ct ohs`	Stops the Oracle HTTP Server processes in the local instance.

To start, stop, and restart HTTP Server processes in clustered environments, the command must include cluster and/or instance options to specify the target OHS processes. For example:

```
dcmctl start -cl myCluster -i myInstance -ct ohs
```

> **See Also:** *Oracle9i Application Server Administrator’s Guide* for more information about clustered environments and DCM.

Starting the Oracle HTTP Server

To start the Oracle HTTP Server, use the start command:

ORACLE_HOME/dcm/bin>dcmctl start -ct ohs (UNIX)

ORACLE_HOME\dcm\bin>dcmctl start -ct ohs (Windows)

Stopping the Oracle HTTP Server

To stop the Oracle HTTP Server, use the stop command:

ORACLE_HOME/dcm/bin>dcmctl stop -ct ohs (UNIX)

ORACLE_HOME\dcm\bin>dcmctl stop -ct ohs (Windows)

This command sends a TERM signal to the parent process, causing it to terminate all of the child processes (which could take several seconds). After all of the children are terminated, the parent exits. Any client requests in progress are terminated, and no other requests are served until the server is started again.

Restarting the Oracle HTTP Server

Restarting the Oracle HTTP Server performs a graceful restart, which is invisible to clients. In a graceful restart, a USR1 signal is sent. When the process receives this signal, it tells the children to exit after processing the current request. (Children that are not servicing requests exit immediately.)

The parent re-reads the configuration files and re-opens the log files, replacing the children with new children in accordance with the settings it finds when re-reading the configuration files. It always observes the process creation settings (`MaxClients`, `MaxSpareServers`, `MinSpareServers`) specified, and takes the current server load into account.

To restart the Oracle HTTP Server, use the `restart` command:

ORACLE_HOME/dcm/bin>`dcmctl restart -ct ohs` (UNIX)

ORACLE_HOME\dcm\bin>`dcmctl restart -ct ohs` (Windows)

2

Specifying the Server and File Locations

This chapter introduces you to the Oracle HTTP Server configuration files, explains how to set Oracle HTTP Server and server administrator options, and specifies file locations in the following topics:

- Accessing Configuration Files
- Setting Server and Administrator Functions
- Specifying File Locations

Note: Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click on the hyperlinks.

Accessing Configuration Files

Oracle HTTP Server is configured by placing *directives*, which are basically instructions, into text configuration files. The configuration files are located in `ORACLE_HOME/Apache/Apache/conf` for UNIX and `ORACLE_HOME\Apache\Apache\conf` for Windows. Some of these files are read only once when the server starts or is reloaded, whereas some files are read every time a related file or directory is requested.

The configuration files which are read only once are called *server-wide* configuration files.

httpd.conf

This is a server configuration file which typically contains directives that affect how the server runs, such as user and group IDs it should use, location of other files. Because the server configuration file is the main file that the server starts with, Oracle HTTP Server doesn't include any directive that says where to locate it. The location is passed on command line when the server starts.

Setting Server and Administrator Functions

You can use the following directives to set basic Oracle HTTP Server and administrator functions:

- ServerName
- UseCanonicalName
- ServerAdmin
- ServerSignature
- ServerTokens
- ServerAlias

ServerName

This enables the server to set a hostname that can be used to create redirection URLs, through which users can access directories without having to use a "/" at the end.

Oracle9*i*AS Web Cache on a Different Machine than Oracle HTTP Server

This section provides information about modifying `ServerName` directive for deployment if Oracle9*i*AS Web Cache is on a different machine than Oracle HTTP Server.

At installation time, Oracle HTTP Server sets the `httpd.conf` file with the following directives that impact Oracle9*i*AS Web Cache:

- `Port=web_cache_port` specifies the Oracle9*i*AS Web Cache listening ports
- `Listen=Oracle_HTTP_Server_port` specifies the HTTP and HTTPS ports obtained by Oracle HTTP Server.
- `ServerName` specifies the host name of Oracle HTTP Server.
- `UseCanonicalName On` instructs Oracle HTTP Server to use the host names and port values set in the `ServerName` and `Port` directives when redirecting a URL.

For example,

```
##
## httpd.conf -- Apache HTTP Server configuration file
##
...
Port 7777
Listen 7778
...
ServerName http_server.company.com
...
UseCanonicalName On
....
```

If Oracle9*i*AS Web Cache is deployed on a separate machine from Oracle HTTP Server, then the Oracle HTTP Server administrator must modify the `ServerName` directive in `httpd.conf` for each site hosted by Oracle9*i*AS Web Cache. This will enable Oracle HTTP Server to redirect URLs to Oracle9*i*AS Web Cache. The following example shows `httpd.conf` modified to set requests for www.1st.company.com and www.2nd.company.com to Oracle9*i*AS Web Cache with a listening port of 7777.

```
Port 7777
Listen 7778
...
ServerName www.1st.company.com
ServerName www.2nd.company.com
...
UseCanonicalName On
....
```

See Also: "ServerName directive" in the Apache Server documentation

UseCanonicalName

This determines which hostname and port to use when redirecting the URL to the same server.

- on: This is the default setting. For this setting, the server uses the hostname and port values set in ServerName and Port.
- off: For this setting, the server uses the hostname and port that the user specifies in the request.

See Also: "UseCanonicalName directive" in the Apache Server documentation

ServerAdmin

This creates an email address that is included with every error message that clients encounter. It is useful to create a separate email address for this.

See Also: "ServerAdmin directive" in the Apache Server documentation

ServerSignature

This enables the server to recognize which server, amongst the various proxies, created the returned response, such as an error message.

- on: This is the default. For this setting, it creates a footer to the returned document that includes information such as ServerName and server version number.
- `email:` For this setting, it additionally creates a "mailto:" reference to the ServerAdmin of the document.
- `off`: For this setting, the footer and mailto: reference is not created.

See Also: "ServerSignature directive" in the Apache Server documentation

ServerTokens

This controls the server information which is returned to clients, such as in error messages. This information includes a description of the generic OS-type of the server and information about compiled-in modules.

- `min(imal)`: For this setting, the server provides information such as server name and version.
- `OS`: For this setting, the server provides information such as server name, version and operating system.
- `full`: For this setting, the server provides information such as server name, version, operating system, and complied modules.

See Also: "ServerTokens directive" in the Apache Server documentation

ServerAlias

This sets alternate names for the current virtual host.

See Also: "ServerAlias directive" in the Apache Server documentation

Specifying File Locations

You can use the following directives to control the location of various server files:

- CoreDumpDirectory
- DocumentRoot
- ErrorLog
- LockFile
- PidFile
- ScoreBoardFile
- ServerRoot

CoreDumpDirectory

This specifies the directory in which the server dumps core. The default is the ServerRoot directory. This directive is applicable to UNIX only.

> **See Also:** “CoreDumpDirectory directive” in the Apache Server documentation

DocumentRoot

This sets the directory from which httpd will serve files. Unless matched by a directive like `Alias`, the server appends the path from the requested URL to the document root to make the path to the document.

> **See Also:** “DocumentRoot directive” in the Apache Server documentation

ErrorLog

This sets the name of the file to which the server will note any errors it encounters. If the name of the file does not begin with a slash, then it is assumed to be relative to the ServerRoot. If the name of the file begins with a pipe (|), then it is assumed to be a command to spawn to handle the error log.

> **See Also:** “ErrorLog directive” in the Apache Server documentation

LockFile

This sets the path to the lockfile used when Oracle HTTP Server is complied with either `USE_FCNTL_SERIALIZED_ACCEPT` or `USE_FLOCK_SERIALIZED_ACCEPT`. It is recommended that default value be used. The main reason for changing it is if the logs directory is NFS mounted, since the lockfile must be stored on a local disk.

See Also: “LockFile directive” in the Apache Server documentation

PidFile

This enables you to set and change the location of the `PID` file to which the server records the process identification number. If the filename does not begin with a slash (/), then it is assumed to be relative to the ServerRoot.

See Also: “PidFile directive” in the Apache Server documentation

ScoreBoardFile

This is required in some architectures to set a file that the server will use to communicate between the parent and children processes. To verify if your architecture requires a scoreboard file is to run Oracle HTTP Server and see if it creates the file named by the directive. If your architecture requires it then you must ensure that this file is not used at the same time by more than one invocation of the server.

See Also: “ScoreBoardFile directive” in the Apache Server documentation

ServerRoot

This specifies the directory that contains the *conf* and *logs* subdirectories. If the server is started with the `-f` option, then you will have to specify ServerRoot.

See Also: “ServerRoot directive” in the Apache Server documentation

3

Managing Server Processes

This chapter provides an overview of the Oracle HTTP Server processes and provides information on how to regulate, and monitor these processes. Topics include:

- HTTP Server Processing Model
- Limiting the Number of Processes and Connections
- Getting Information about Processes

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click on the hyperlinks.

HTTP Server Processing Model

Once Oracle HTTP Server is started, the system is ready to listen for and respond to http(s) requests. The request processing model is different for Window and UNIX.

On UNIX, when Oracle HTTP Server is started, a single parent process launches several *child* processes that listen and promptly respond to client requests. The main httpd parent process continues to run as the root user if the `root.sh` script was run during installation or if the user was logged in as `root`. However, the child processes run as a less privileged user. The User and Group directive are used to set the privileges for the child processes. The child processes must be able to read all the content that will be served.

On Windows, Oracle HTTP Server launches a single parent process and one child process. The child process creates multiple threads that listen and respond to client requests.

You must decide how you want to set Oracle HTTP Server to handle processes.

ServerType

This directive provides two options for this:

- `inetd:` This starts up a new child process every time a request comes in. The program exits once the request is dealt with. This setting eliminates the option of having several child processes in waiting. It can be slower and expensive, but more secure. This is applicable to UNIX only.
- `standalone:` This enables several waiting child processes and requires the server to be started only once. This is the default, and recommended setting for a busy Web site. This is applicable to UNIX only.

> **See Also:** “ServerType directive” in the Apache Server documentation

You must specify the `Group` and `User` under which the server will answer requests. This is applicable to UNIX only.

Group

This specifies the group under which the server will answer requests. In order to use this directive, the standalone server must be run initially as root. It is recommended that you create a new group for running the server.

See Also: "Group directive" in the Apache Server documentation

User

This specifies the userid to which the server will answer requests. In order to use this directive, the standalone server must be run initially as root. The user should have privileges to access files that are available for everyone, and the user should not be able to execute code which is not meant for httpd requests. It is recommended that you set up a new user for running the server.

See Also: "User directive" in the Apache Server documentation

Limiting the Number of Processes and Connections

The following directives control and limit the number of child processes or simultaneous requests:

- StartServers
- ThreadsPerChild
- MaxClients
- MaxRequestPerChild
- MaxSpareServers
- MinSpareServers

StartServers

This sets the number of child server processes created when Oracle HTTP Server is started. The default is set at 5. This is applicable to UNIX only.

See Also: "StartServers directive" in the Apache Server documentation

ThreadsPerChild

This controls the maximum number of child threads handling requests. This is applicable to Windows only.

See Also: "ThreadsPerChild directive" in the Apache Server documentation

MaxClients

This limits the number of requests that can be dealt with at one time. The default and recommended value is 150. This is applicable to UNIX only.

See Also: "MaxClients directive" in the Apache Server documentation

MaxRequestPerChild

This controls the number of requests a child process handles before it dies. This value should be specified again if the machine is rebooted. If you select the value to be 0, which is the default, then the process will never die. This is applicable to UNIX only.

See Also: "MaxRequestPerChild directive" in the Apache Server documentation

MaxSpareServers

This sets the maximum number of idle child server processes. An idle process is one which is running but not handling a request. The parent process will kill off idle child processes that exceed the value set for this directive. The default is set at 10. This is applicable to UNIX only.

See Also: "MaxSpareServers directive" in the Apache Server documentation

MinSpareServers

This sets the minimum number of idle child server processes. An idle process is one which is running but not handling a request. The parent process will create new children at the maximum rate of one process per second if there are fewer processes running. The default is set at 5. This is applicable to UNIX only.

> **See Also:** "MinSpareServers directive" in the Apache Server documentation

Getting Information about Processes

To monitor HTTP Server processes, you can use the performance monitor on Windows, or the `ps` utility on UNIX.

> **See Also:** *Oracle9i Application Server Performance Guide* and your operating system documentation for more information.

You can also monitor the HTTP Server processes using the Oracle Enterprise Manager Oracle9*i*AS Home Page.

> **See Also:** *Oracle9i Application Server Administrator's Guide*

If a network error occurs on a device such as a router or firewall between the application server and the database, JDBC connections may stop responding. In this situation, you must stop the HTTP Server and JServ processes manually, and there may be a delay in stopping the processes.

4

Managing the Network Connection

This chapter provides information about specifying IP addresses and ports, and managing server interaction and network connection persistence. Topics include:

- Specifying Listener Ports and Addresses
- Managing Interaction between the Server and Network
- Managing Connection Persistence

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click on the hyperlinks.

Specifying Listener Ports and Addresses

When Oracle HTTP Server is started, by default it listens for requests on port 7777 (non-SSL) or 4443 (SSL). For non-SSL, if port 7777 is occupied, Oracle HTTP Server will listen on the next available port number between a range of 7777-7877. Thus, if port 7777 is busy, it would listen on port 7778 and so on. Similarly, for SSL, if port 4443 is occupied, it will listen on the next available port number between the range of 4443-4543. Thus, if 4443 is busy, it will listen on 4444 and so on.

A file named `setupinfo.txt` is automatically generated in *`ORACLE_HOME`*`/Apache/Apache.` It contains information about which port Oracle HTTP Server is listening on. This file is generated at install time, and is not updated thereafter. If you restart Oracle HTTP Server, the information in `setupinfo.txt` becomes inaccurate.

Users can specify the server to listen to more than one port, selected addresses, or a combination. The following directives specify listener ports and addresses. Note that `BindAddress` and `Port` directives can be used only once. Apache group recommends the use of the `Listen` directive instead.

- BindAddress
- Listen
- Port

BindAddress

This restricts the server to listen to a single IP address. If the argument to this directive is *, then it will listen to all IP addresses.

See Also: “BindAddress directive” in the Apache Server documentation

Port

If no `Listen` or `BindAddress` directives are present, then this directive specifies the port of the listener. If a `Listen` directive is present, the `Port` value becomes the default port value that will be used when Oracle HTTP Server builds URLs or other references to itself. Usually, the values of `Port` and `Listen` should match, unless Oracle HTTP Server is being fronted by a caching or proxy server. In this case, you may want to set `Port` to be the port that is being used by the front end server and `Listen` to the port that Oracle HTTP Server is actually listening to. By doing this, redirects or other URLs generated by Oracle HTTP Server will point to the front end server rather than directly to Oracle HTTP Server.

> **See Also:** “Port directive” in the Apache Server documentation

Listen

This specifies an IP port that Oracle HTTP Server should listen on. Multiple `Listen` directives can be used to listen on multiple ports. If present, this value will override the value of `Port`. Accordingly, if you have a `Port` value of 7777 and a `Listen` value of 7778, then Oracle HTTP Server will only listen on one port, 7778.

> **See Also:** “Listen directive” in the Apache Server documentation

Managing Interaction between the Server and Network

The following directives are used to specify how the server interacts with the network:

- ListenBackLog
- SendBufferSize
- TimeOut

ListenBackLog

This specifies the maximum length of the queue of pending connections. This is useful if the server is experiencing a TCP SYN overload, which causes numerous new connections that open up but don’t complete the task.

> **See Also:** “ListenBackLog directive” in the Apache Server documentation

SendBufferSize

This increases the TCP buffer size to the number of bytes specified, thereby improving performance.

> **See Also:** “SendBufferSize directive” in the Apache Server documentation

TimeOut

This sets the maximum time, in seconds, that the server waits for the following:

- The total amount of time it takes to receive a GET request.
- The amount of time between receipt of TCP packets on a POST or PUT request.
- The amount of time between ACKs on transmissions of TCP packets in responses.

The default is set at 300 seconds.

See Also: "TimeOut directive" in the Apache Server documentation

Managing Connection Persistence

The following directives configure how the server handles persistent connections.

- KeepAlive
- KeepAliveTimeout
- MaxKeepAliveRequests

KeepAlive

This enables a connection to be open for a long time, which enables multiple requests to be sent over the same TCP connection. The default is set to "ON".

See Also: "KeepAlive directive" in the Apache Server documentation

KeepAliveTimeout

This sets the number of seconds the server will wait for a subsequent request before closing the connection. Once a request has been received, the timeout value specified by the TimeOut directive applies. The default is set at 15 seconds.

See Also: "KeepAliveTimeout directive" in the Apache Server documentation

MaxKeepAliveRequests

This limits the number of requests allowed per connection when KeepAlive is on. If it is set to “0”, unlimited requests will be allowed. The default is set at 100.

See Also: “MaxKeepAliveRequests directive” in the Apache Server documentation

5

Configuring and Using Server Logs

This chapter discusses the log formats and describes various log files and their locations. Topics include:

- Specifying the Log Formats
- Specifying Log Files and Locations

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click on the hyperlinks.

Specifying the Log Formats

`LogFormat` specifies the information included in the log file, and the manner in which it is written. The default format is the Common Log Format. The CLF format is: `host ident authuser date request status bytes`

`host`: This is the client domain name or its IP number.

`ident`: If `IdentityCheck` is enabled and the client machine runs `identd`, then this is the client identity information.

`authuser`: This is the user ID for a password-protected site.

`date`: This is the date and time of the request in the `<day/month/year:hour:minute:second>` format.

`request`: This is the request line, in double quotes, from the client.

`status`: This is the three-digit status code returned to the client.

`bytes`: This is the number of bytes, excluding headers, returned to the client.

Specifying Log Files and Locations

The following section describes the function and location of log files.

Access Log

The server access log records all requests processed by the server. The location and content of the access log is controlled by the `CustomLog` directive. The `LogFormat` directive can be used to simplify the selection of the contents of the logs.

Note: You can integrate Oracle9*i*AS Web Cache access logs into Oracle9*i*AS Clickstream Intelligence with the Collector Agent. See the *Oracle9iAS Clickstream Intelligence Administrator's Guide* for details.

See Also: "AccessLog directive" in the Apache Server documentation

Error Log

The server records error messages to a log file located, by default, in *ORACLE_HOME*/`Apache/Apache/logs/error_log` in UNIX, and *ORACLE_HOME*`\Apache\Apache\logs\error_log` in Windows. The filename can be set using the `ErrorLog` directive.

See Also: "ErrorLog directive" in the Apache Server documentation

JServ Log

JServ log tracks actions performed, and exceptions generated from JServ applications, such as servlets and JSPs. It is located in *ORACLE_HOME*/`Apache/Jserv/logs/jserv.log` in UNIX, and *ORACLE_HOME*`\Apache\Jserv\logs\jserv.log` in Windows.

PID File

When the server is started, it notes the process id of the parent httpd process to the PID file located by, default, in *ORACLE_HOME*/`Apache/Apache/logs/httpd.pid`. This filename can be changed with the `PidFile` directive. The process id is for use by the administrator in restarting and terminating the daemon. If the process dies (or is killed) abnormally, then it will be necessary to kill the children httpd processes.

See Also: "PidFile directive" in the Apache Server documentation

Rewrite Log

Rewrite Log is necessary for debugging when `mod_rewrite` is used. This log file produces a detailed analysis of how the rewriting engine transforms requests. The level of detail is controlled by the `RewriteLogLevel` directive.

See Also: "RewriteLog directive" in the Apache Server documentation

Script Log

Script Log allows you to record the input to and output from the CGI scripts. This should only be used in testing, and not for live servers.

> **See Also:** “ScriptLog directive” in the Apache Server documentation

SSL Log

When Oracle HTTP Server starts in SSL mode, it creates `ssl_engine_log` and `ssl_requrest_log` in *`ORACLE_HOME/`*`Apache/Apache/logs` in UNIX, and *`ORACLE_HOME`*`\Apache\Apache\logs` in Windows. `ssl_engine_log` tracks ssl and protocol issues, where as `ssl_request_log` records user activity. Use the `SSLLogFile` directive to control output.

> **Note:** On Windows, Oracle HTTP Server is starts in SSL mode by default.

Transfer Log

Transfer Log specifies the file in which to store the log of accesses to the site. If it is not explicitly included in the *conf* file, then no log is generated. The server will typically log each request to a transfer file located, by default, in *`ORACLE_HOME/`*`Apache/Apache/logs/access_log` in UNIX, and *`ORACLE_HOME`*`\Apache\Apache\logs\access_log` in Windows. The filename can be set using a `CustomLog` directive.

6

Oracle HTTP Server Modules

This chapter describes the modules (mods) included in the Oracle HTTP Server. Documentation from the Apache Software Foundation is referenced when applicable.

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click on the hyperlinks.

mod_access

This module controls access to the server based on characteristics of a request, such as hostname or IP address.

See Also: Module mod_access in the Apache Server documentation

This module is available for UNIX systems only.

mod_actions

This module enables execution of CGI scripts based on file type or request method.

See Also: Module mod_actions in the Apache Server documentation

mod_alias

This module enables manipulation of URLs in processing requests. It provides mapping between URLs and filesystem paths, and URL redirection capabilities.

See Also: Module mod_alias in the Apache Server documentation

mod_asis

This module enables sending files that contain their own HTTP headers.

This module is not supported by Oracle.

mod_auth

This module enables user authentication with text files.

See Also: Module mod_auth in the Apache Server documentation

mod_auth_anon

This module enables anonymous user access to protected areas (similar to anonymous FTP, where the email addresses can be logged).

> **See Also:** Module mod_auth_anon in the Apache Server documentation

mod_auth_db

This module uses Berkeley DB files to provide user authentication.

This module is disabled in the Oracle HTTP Server and is not supported by Oracle.

mod_auth_dbm

This module uses DBM files to provide user authentication.

This module is not supported by Oracle.

mod_auth_digest

This module uses MD5 Digest Authentication to provide user authentication.

This module is not supported by Oracle.

mod_autoindex

This module generates directory indexes automatically.

> **See Also:** Module mod_autoindex in the Apache Server documentation

mod_cern_meta

This module emulates CERN (Conseil Europeen pour le Recherche Nucleaire) HTTPD metafile semantics. Metafiles are additional HTTP headers that can be produced for each file the server accesses, in addition to the typical set.

This module is not supported by Oracle.

mod_cgi

This module enables the server to run CGI scripts.

See Also: Module mod_cgi in the Apache Server documentation

A demonstration of CGI capabilities is available from the Oracle9*i*AS Welcome page. Click the Demonstrations tab, then the J2EE and Web Cache link.

mod_define

This module enables the Define directive, which defines a variable that can be expanded on any configuration line. The Define directive has the status Extension, which means that it is not compiled into the server by default.

This module requires the Extended API (EAPI).

This module is available for UNIX systems only.

mod_digest

This module uses an older version of the MD5 Digest Authentication specification than that used in mod_auth_digest to provide user authentication. mod_digest probably only works with older browsers.

See Also: Module mod_digest in the Apache Server documentation

mod_dir

This module enables the server to perform “trailing slash” redirects. Directories must contain a trailing slash. If a request for a URL without a trailing slash is received, mod_dir redirects the request to the same URL followed by a trailing slash. For example:

```
http://myserver/documents/mydirectory
```

is redirected to

```
http://myserver/documents/mydirectory/
```

See Also: Module mod_dir in the Apache Server documentation

mod_dms

This module enables you to monitor performance of site components with Oracle's Dynamic Monitoring Service.

> **See Also:** *Oracle9i Application Server Performance Guide*.

mod_env

This module enables you to control the environment for CGI scripts and SSI (Server Side Include) pages by passing, setting, and unsetting environment variables.

> **See Also:** Module mod_env in the Apache Server documentation

mod_example

This module provides examples and guidance on how to write modules using the Apache API. When implemented, it demonstrates module callbacks triggered by the server.

This module is not supported by Oracle.

mod_expires

This module enables the server to generate Expires HTTP headers, which provide information to the client about document validity. Documents are served from the source if, based on the expiration criteria, the cached copy has expired.

> **See Also:** Module mod_expires in the Apache Server documentation

mod_fastcgi

This third-party module supports the fastcgi protocol, which enables you to maintain a pool of running servers for CGI applications (thereby eliminating start-up and initialization overhead).

A demonstration of FastCGI capabilities is available from the Oracle9*i*AS Welcome page. Click the Demonstrations tab, then the J2EE and Web Cache link.

Note: After installation of Oracle9*i*AS, error messages for FastCGI may appear in the Oracle HTTP Server error log (ORACLE_HOME/Apache/Apache/logs). The messages indicate a server access failure and read by group not allowed. These messages do not affect the operation of FastCGI. They are caused by re-ordering of directives in `httpd.conf` during installation. To eliminate the messages, move the User and Group directives from the end of the file to a location preceding the FastCGI server directives.

Warning: The demonstration script for this module that is shipped with Oracle9*i*AS should be disabled in production environments. It is included only to verify that the installation was successful.

See Also: Module mod_fastcgi in the Apache Server documentation

mod_headers

This module enables you to merge, replace, or remove HTTP response headers.

See Also: Module mod_headers in the Apache Server documentation

mod_imap

This module enables server-side image map processing.

This module is not supported by Oracle.

mod_include

This module provides a filter that processes documents for SSI (Server Side Include) directives.

See Also: Module mod_include in the Apache Server documentation

mod_info

This module summarizes the entire server configuration, including all installed modules and directive settings.

> **See Also:** Module mod_info in the Apache Server documentation

mod_isapi

This module is available on the Windows platform only. It enables serving of Internet Server extensions (such as.dll modules).

This module is not supported by Oracle.

mod_jserv

This module connects the Oracle HTTP Server to the JServ servlet engine. It converts HTTP requests to servlet requests, returning HTTP responses to the client.

mod_jserv is disabled by default in the Oracle HTTP Server distribution; it is included for legacy support only. The instructions below explain how to enable it with mod_oprocmgr, in manual mode, or in automatic mode. Use the instructions for the mode that serves your needs. A working knowledge of JServ and Oracle HTTP Server directives is assumed.

Enabling JServ with mod_oprocmgr

This section explains how to enable the Oracle default mode for JServ. Use this mode if you want process management and load balancing capabilities for multiple JVMs. The ApJServManual directive has a new mode, ’auto’, that enables using JServ with the Oracle module mod_oprocmgr. The file `jserv.conf` file contains LoadModule directives for mod_jserv and mod_oprocmgr.

Follow these steps to enable JServ with mod_oprocmgr:

1. Uncomment the Include directive for the `jserv.conf` file in:

   `ORACLE_HOME``/Apache/Apache/conf/httpd.conf` (UNIX)

   ```
   #include "/ORACLE_HOME/Apache/Jserv/etc/jserv.conf"
   ```

   `ORACLE_HOME``\Apache\Apache\conf\httpd.conf` (Windows)

   ```
   #include "C:\ORACLE_HOME\Apache\Jserv\conf\jserv.conf"
   ```

2. Configure directives, if needed, in the file:

   ORACLE_HOME/Apache/Jserv/etc/jserv.conf (UNIX)

   ORACLE_HOME\Apache\Jserv\conf\jserv.conf (Windows)

3. Configure directives, if needed, in the file:

   ORACLE_HOME/Apache/Jserv/etc/jserv.properties (UNIX)

   ORACLE_HOME\Apache\Jserv\conf\jserv.properties (Windows)

4. Configure directives, if needed, in the file:

   ORACLE_HOME/Apache/Jserv/etc/zone.properties (UNIX)

   ORACLE_HOME\Apache\Jserv\conf\zone.properties (Windows)

5. Configure JServ using the Enterprise Manger Web site:

   a. Navigate to the Instance Home Page on the Enterprise Manager Web site. Scroll to the Administration section.

   b. Select **Configure Components**. This opens the Configure Components Page.

   c. Choose JServ in the Component drop-down menu, enter the ias_admin password, and click **OK**.

6. Restart the Oracle HTTP Server.

Enabling JServ in Automatic Mode

This section explains how to enable JServ in automatic mode. Use this mode if you need only one JVM. In this mode, the ApJServManual directive is set to ’off’ and the mod_jserv module launches and monitors the JVM. If the Oracle HTTP Server is restarted or stopped, mod_jserv restarts or stops the JVM.

Follow these steps to enable JServ in automatic mode:

1. Uncomment the Include directive for the jserv.conf file in:

   ORACLE_HOME/Apache/Apache/conf/httpd.conf (UNIX)

   ```
   #include "/ ORACLE_HOME/Apache/Jserv/etc/jserv.conf"
   ```

   ORACLE_HOME\Apache\Apache\conf\httpd.conf (Windows)

   ```
   #include "C:\ORACLE_HOME\Apache\Jserv\conf\jserv.conf"
   ```

2. Configure the ApJServManual directive in the file:

   `ORACLE_HOME/Apache/Jserv/etc/jserv.conf` (UNIX)

   `ORACLE_HOME\Apache\Jserv\conf\jserv.conf` (Windows)

   ```
   ApJServManual off
   ```

3. Configure other directives as needed in `jserv.conf`.
4. Set the port directive in the file:

   `ORACLE_HOME/Apache/Jserv/etc/jserv.properties` (UNIX)

   `ORACLE_HOME\Apache\Jserv\conf\jserv.properties` (Windows)

   to the same value as that specified in the ApJServDefaultPort directive.
5. Configure directives, if needed, in the file:

   `ORACLE_HOME/Apache/Jserv/etc/zone.properties` (UNIX)

   `ORACLE_HOME\Apache\Jserv\conf\zone.properties` (Windows)
6. Configure JServ using the Enterprise Manger Web site:

   a. Navigate to the Instance Home Page on the Enterprise Manager Web site. Scroll to the Administration section.

   b. Select **Configure Components**. This opens the Configure Components Page.

   c. Choose JServ in the Component drop-down menu, enter the `ias_admin` password, and click **OK**.
7. Restart the Oracle HTTP Server.

Enabling JServ in Manual Mode

This section explains how to enable JServ in manual mode. Use this mode if you need to run multiple JVMs. In this mode, the ApJServManual directive is set to ’on’ and you have to stop and start the JVM manually. To monitor the JVM, you must use an external monitoring facility.

Follow these steps to enable JServ in manual mode:

1. Uncomment the Include directive for the jserv.conf file in the file:

   ORACLE_HOME/Apache/Apache/conf/httpd.conf (UNIX)

   ```
   #include "/ ORACLE_HOME/Apache/Jserv/etc/jserv.conf"
   ```

   ORACLE_HOME\Apache\Apache\conf\httpd.conf (Windows)

   ```
   #include "C:\ORACLE_HOME\Apache\Jserv\conf\jserv.conf"
   ```

2. Configure the ApJServManual directive in the file:

   ORACLE_HOME/Apache/Jserv/etc/jserv.conf (UNIX)

   ORACLE_HOME\Apache\Jserv\conf\jserv.conf (Windows)

   ```
   ApJServManual on
   ```

3. Configure other directives as needed in jserv.conf.
4. Configure directives in the file:

   ORACLE_HOME/Apache/Jserv/etc/jserv.properties (UNIX)

   ORACLE_HOME\Apache\Jserv\conf\jserv.properties (Windows)

5. Configure directives in the file:

   ORACLE_HOME/Apache/Jserv/etc/zone.properties (UNIX)

   ORACLE_HOME\Apache\Jserv\conf\zone.properties (Windows)

6. Before or while starting the JVM, set the arguments passed to the Java interpreter, and the classpath passed to the JVM (as specified by wrapper.bin.parameters and wrapper.classpath in the jserv.properties file).

> **Note:** Scripts are provided in the ORACLE_HOME/Apache/Apache/bin directory to start and stop JServ. These include commands to set the arguments and the classpath.

7. Configure JServ using the Enterprise Manger Web site:

   a. Navigate to the Instance Home Page on the Enterprise Manager Web site. Scroll to the Administration section.

   b. Select **Configure Components**. This opens the Configure Components Page.

   c. Choose JServ in the Component drop-down menu, enter the `ias_admin` password, and click **OK**.

8. Restart the Oracle HTTP Server.

Using JServ and OC4J Together

This section explains how to use mod_rewrite to enable some applications to execute on JServ, and others on OC4J.

Perform the following configuration steps to enable JServ and Oracle9iAS Containers for J2EE (OC4J) to coexist. This is important if you have the Portal and Wireless installation type, because of the Portal dependency on OC4J.

1. Specify the engine on which applications should execute. Suppose you have these URLs:

   `/application1/file1.jsp` to execute on JServ, and

   `/application2/file2.jsp` to execute on OC4J.

   You must rewrite the URL for application1.

   a. Edit:

      `ORACLE_HOME``/Apache/Apache/conf/httpd.conf` (UNIX)

      `ORACLE_HOME``\Apache\Apache\conf\httpd.conf` (Windows)

      and ensure that the following directives are present and active (uncommented):

      ```
      LoadModule rewrite_module libexec/mod_rewrite.so
      AddModule mod_rewrite.c
      RewriteEngine on
      ```

b. Edit:

ORACLE_HOME/Apache/jsp/conf/ojsp.conf (UNIX)

ORACLE_HOME\Apache\jsp\conf\ojsp.conf (Windows)

to add these directives:

```
RewriteRule /application1/(.*)/(.*)˙jsp$ /application1/$1/$2.jsp1
ApJServAction .jsp1 /servlets/oracle.jsp.JspServlet
```

c. Remove this directive:

```
ApJServAction .jsp /servlets/oracle.jsp.JspServlet
```

d. Edit:

ORACLE_HOME/Apache/Jserv/etc/jserv.conf (UNIX)

ORACLE_HOME\Apache\Jserv\conf\jserv.conf (Windows)

and mount /servlets to the JVM that will service the JSP requests. Use the ApJServMount or ApJServGroupMount directive (depending on how the JServ processes are started).

2. Configure JServ using the Enterprise Manger Web site:

a. Navigate to the Instance Home Page on the Enterprise Manager Web site. Scroll to the Administration section.

b. Select Configure Components. This opens the Configure Components Page.

c. Choose JServ in the Component menu, enter the ias_admin password, and click OK.

3. Restart the Oracle HTTP Server.

See Also: JServ in the Apache Server documentation

Configuring Multiple JSP Applications on Different JVMs with mod_jserv

mod_jserv's mapping for JSP applications does not provide for specifying application paths, such as:

```
ApJServAction /path/.jsp ...
```

However, you can configure different JSP applications to run on different JVMs under mod_jserv. The configuration steps below show how to use mod_rewrite to change the extension of ' JSP pages associated with a JSP application at request time (where `*.jsp1` files belong to `application1`, and `*.jsp2` files belong to `application2`). Each `.jsp` extension has its own `ApJServAction` handler, so that multiple JVMs can be used to run different JSP applications.

Follow the instructions below, substituting application names, directories, page extensions, and hostnames as applicable to your system:

1. Enable mod_rewrite by adding the following lines to httpd.conf:

   ```
   LoadModule rewrite_module libexec/mod_rewrite.so
   AddModule mod_rewrite.c
   RewriteEngine on
   ```

2. Set up the applications as follows in `ojsp.conf`:

   ```
   RewriteRule /app1/(.*)/(.*)˙jsp$ /app1/$1/$2.jsp1
   RewriteRule /app2/(.*)/(.*)˙jsp$ /app2/$1/$2.jsp2

   ApJServAction .jsp1 /servlets1/oracle.jsp.JspServlet
   ApJServAction .jsp2 /servlets2/oracle.jsp.JspServlet
   ```

3. Mount /servlets1 and /servlets2 to different JVMs in `jserv.conf`:

   ```
   ApJServMount /servlets1
   ajpv12://hostname:8008/root
   ApJServMount /servlets2
   ajpv12://hostname:8009/root
   ```

mod_log_agent

This module enables logging of client user agents. It is deprecated; you should use mod_log_config instead of mod_log_agent.

This module is not supported by Oracle.

mod_log_config

This module provides configurable, customizable logging of server activities. You can choose the log format, and select or exclude individual requests for logging, based on characteristics of the requests.

> **See Also:** Module mod_log_config in the Apache Server documentation

mod_log_referer

This module enables logging of documents that reference documents on the server. It is deprecated; you should use mod_log_config instead of mod_log_referer.

> **See Also:** Module mod_log_referer in the Apache Server documentation

mod_mime

This module enables the server to determine the type of a file from its filename, and associate files with handlers for processing.

> **See Also:** Module mod_mime in the Apache Server documentation

mod_mime_magic

This module enables the server to determine the MIME type of a file by examining a few bytes of its content. It is used in cases when mod_mime cannot determine a file type. Make sure that mod_mime appears before mod_mime_magic in the configuration file, so that mod_mime processes the files first.

> **See Also:** Module mod_mime_magic in the Apache Server documentation

mod_mmap_static

This module maps a list of files into memory, useful for frequently requested files that are not changed often.

This module is not supported by Oracle.

mod_negotiation

This module enables the server for content negotiation (selection of documents based on the client's capabilities).

> **See Also:** Module mod_negotiation in the Apache Server documentation

mod_oc4j

This Oracle module routes requests from the Oracle HTTP Server to Oracle9*i*AS Containers for J2EE (OC4J), providing the ajp13 protocol for communication with the servlet engine.

> **See Also:** *Oracle9iAS Containers for J2EE User's Guide*.

mod_oc4j is enabled by default. During installation, the oc4j_deploy_tool.jar adds mount points to mod_oc4j.conf for applications deployed into OC4J instances. Requests that come in for specific mount points in mod_oc4j are routed to the OC4J instance for that mount point.

OC4J instances are started and managed by Oracle Process Management and Notification (OPMN). OPMN is briefly described in Chapter 1, "Overview" in the section "Starting, Stopping, and Restarting the Oracle HTTP Server". See the *Oracle9i Application Server Administrator's Guide* for detailed information on OPMN.

Security Considerations for mod_oc4j

Be aware of the following security considerations when using mod_oc4j:

- mod_oc4j communicates with OC4J using the ajp13 protocol, which is not SSL-enabled. For this reason, you should run the Oracle HTTP Server and OC4J processes inside a firewall. If there is a firewall between the Oracle HTTP Server and OC4J processes, the ports bound by the OC4J ajp13 listener must be open.
- If configured, mod_oc4j passes some security environment parameters (see Oc4jExtractSSL and Oc4jEnvVar on page 6-20) to OC4J, set by mod_ossl and mod_osso, at request time.

Configuring mod_oc4j

All relevant directives in `httpd.conf` and `mod_oc4j.conf` are described below. Sample configurations are also provided.

mod_oc4j Configuration File

The mod_oc4j directives are maintained in their own file, `mod_oc4j.conf`. The `mod_oc4j.conf` file is included by default into the `httpd.conf` file, using the directive below:

```
include "ORACLE_HOME/Apache/Apache/conf/mod_oc4j.conf"
```

where *ORACLE_HOME* is the path to the Oracle home in which the HTTP Server resides.

mod_oc4j Directives

The directives you use to configure mod_oc4j are described below.

LoadModule

This directive loads the mod_oc4j module. It is included in the default configuration file, so you

Syntax:	LoadModule oc4j_module *mod_oc4j shared library file*
Required:	Yes
Default:	None (UNIX) LoadModule oc4j_module modules/ApacheModuleOc4j.dll (Windows)
Example:	`LoadModule oc4j_module mod_oc4j.so` (UNIX) LoadModule oc4j_module modules/ApacheModuleOc4j.dll (Windows)

Oc4jMount

This directive tells mod_oc4j to route requests containing a particular path to a destination. A destination can be a single OC4J process or a set of OC4J instances.

Syntax: Oc4jMount *path* [*destination*]

where path is the context root. The path parameter must be the same as the application context root specified in the OC4J configuration file `xxx-web-site.xml`. The application context root is shown in bold text in the example <web-site> element below.

<default-web-app application="default" name="defaultWebApp" root="**/j2ee**"/>

and where destination is one of these types:

ajp13_dest

cluster_dest (this is the default destination type)

instance_dest

If destination is not specified, the default OC4J instance name of `home` will be used. For example,

```
Oc4jMount /myApp/*
```

provides the same result as:

```
Oc4jMount /myApp/* cluster://local_ias_cluster_
name:home
```

Required: No

Default: None

Examples: Oc4jMount /app01/* ajp13://my-sun:8888

Oc4jMount /app02/*

Oc4jMount /app03/* home

Oc4jMount /app04/* ias_cluster_1:home

Oc4jMount /app05/* cluster://ias_cluster_1:home,ias_cluster_2:home

Oc4jMount /app06/* instance://ias_instance_1:home

Oc4jMount /app07/* instance://ias_instance_1:home_1,ias_instance_2:home_2

Oc4jMount /app08/* instance://my-sun:ias_instance_1:home

Usage: Examples are provided for each routing destination:

ajp_13 dest

Oc4jMount *path* ajp13://*my-sun:8888*

A request with the pattern specified in *path* is routed to an OC4J process listening on my-sun, port 8888 with the ajp13 protocol. (my-sun and port 8888 are the ajp13 protocol host and port specified in the OC4J configuration file *xxx*-web-site.xml.

cluster_dest

Oc4jMount *path* cluster://*iAS Cluster Name*:*OC4J Instance Name*, *iAS Cluster Name*:*OC4J Instance Name*...

A request with the pattern specified in *path* is load balanced to one or more of the OC4J instances specified (instances are separated by commas).

The iAS Cluster Name is optional. If it is provided, the destination OC4J instance should be inside the named cluster. If none is provided, the destination OC4J instance should be inside the local iAS cluster.

instance_dest

Oc4jMount *path* instance://host:*iAS Instance Name*:*OC4J Instance Name*, host:*iAS Instance Name*:*OC4J Instance Name*...

A request with the pattern specified in *path* is load balanced to one or more of the OC4J instances specified (instances are separated by commas).

The host name is optional. If it is provided, the destination OC4J instance should be inside the *i*AS instance residing on that host. If none is provided, the destination OC4J instance could be on any host.

Oc4jMountCopy

This directive copies mount points from the base server.

Syntax:	Oc4jMountCopy *on* | *off*
Required:	No
Default:	on
Example:	`Oc4jMountCopy off`
Usage:	This directive tells mod_oc4j whether to copy Oc4jMount points from the base server to the virtual host on which this directive is specified. If its value is On, all of the Oc4jMount points configured in the base server will be copied to the virtual host. If its value is Off, only the Oc4jMount points configured within the virtual host scope will be used.

Oc4jCacheSize

This directive specifies the size of the OC4J connection cache.

Syntax:	Oc4jCacheSize *size of connection cache*
Required:	No
Default:	UNIX: 1 Windows: 32
Example:	`Oc4jCacheSize 64`
Usage:	This directive specifies the number of concurrent OC4J connections that can be cached by one httpd process. If there are more than 1024 Oracle HTTP Server processes running on the same computer and accessing OC4J instances, you should set this directive to 0. This disables persistent connections between mod_oc4j and the OC4J instances, thereby improving performance.

Oc4jExtractSSL

This directive governs passing SSL environment variables.

Syntax:	Oc4jExtractSSL On | Off
Required:	No
Default:	Off
Example:	Oc4jExtractSSL On
Usage:	This directive tells mod_oc4j whether or not to pass three SSL environment variables, SSL_CLIENT_CERT, SSL_CIPHER, and SSL_SESSION_ID to OC4J. There is a performance cost associated with copying the SSL environment variables to OC4J, so set it to On only if the environment variables must be available to OC4J.

Oc4jEnvVar

This directive tells mod_oc4j to pass some environment variables from the Oracle HTTP Server to OC4J.

Syntax:	Oc4jEnvVar *environment variable name* [*environment variable default value*]
Required:	No
Default:	None
Example:	Oc4jEnvVar MY_ENV1 Oc4jEnvVar MY_ENV2 myenv_value
Usage:	For each OC4jEnvVar entry, you must also configure the Oracle HTTP Server directive, PassEnv, with the environment variable. Otherwise, mod_oc4j cannot acquire and pass the value. Multiple entries are allowed. You could specify the default value for the environment variable as the second parameter, or leave it empty. If the environment variable's value is found in the Oracle HTTP Server environment, its value will be passed to OC4J. Otherwise, if the default value is set, the default value will be passed. If this environment variable's value is not found in the Oracle HTTP Server environment and the default value is not set, nothing is passed to OC4J. There is a performance degradation associated with mod_oc4j passing some configured environment variables over to OC4J with each request.

Sample Configurations for mod_oc4j

This section provides some sample configurations for mod_oc4j.

Level 1 Configuration

Level 1 is the simplest configuration. Two examples of Level 1 configurations are given below.

Example A This configuration mounts all requests starting with the URI /servlet/ to the default instance of OC4J processes. Because a instance of OC4J processes is handled by OPMN and the default instance must be the same as OPMN's default OC4J instance, this configuration requires that mod_oc4j must be used with OPMN.

1. Make this entry in the httpd.conf file:

```
Oc4jMount /servlet/*
```

Example B This configuration performs the same work as the configuration in Example A, using a Location container instead of the Oc4jMount directive.

1. Make this entry in the httpd.conf file:

```
<Location /servlet>
    SetHandler oc4j-handler
</Location>
```

Example C This configuration mounts all requests starting with the URI /servlet/ or /j2ee/ and all JSP pages to the default OC4J instance of OC4J processes. This configuration requires that mod_oc4j must be used with OPMN.

1. Make these entries in the oc4j.conf file:

```
Oc4JMount /servlet/*
Oc4JMount /*.jsp
Oc4JMount /j2ee/*
```

Example D This configuration mounts:

- All requests starting with the URI /applicationA/ and all JSP pages to oc4j_instance_A, in which all OC4J processes are managed by OPMN. This configuration requires that mod_oc4j must be used with OPMN.
- All requests starting with the URI /applicationB/ to oc4j_instance_B, in which all OC4J processes are managed by OPMN. This configuration requires that mod_oc4j must be used with OPMN.

1. Make these entries in the oc4j.conf file:

```
Oc4JMount /applicationA/* oc4j_instance_A
Oc4JMount /applicationB/* oc4j_instance_B
Oc4JMount /j2ee/*
Oc4JMount /*.jsp oc4j_instance_A
```

mod_oprocmgr

This Oracle module provides process management and load balancing services to JServ processes. It is provided for legacy users of JServ. JServ is disabled by default in the Oracle HTTP Server configuration. Oracle Corporation recommends using OC4J and mod_oc4j (which are enabled by default).

Configuring mod_oprocmgr to Provide Process Management and Load Balancing to JServ

This section explains how to configure mod_oprocmgr. Terms used in this section to describe the module and its functions are defined below:

mod_oprocmgr

A module that starts, stops, and detects death of processes (starting new processes to replace them), and provides load balancing services to the processes. mod_oprocmgr gets the topology management information via HTTP requests from JServ, and does its job based on this information.

Group

A set of processes across which request traffic is distributed.

Servlet Engine Process

A JVM instance that runs a servlet engine, such as JServ.

How mod_oprocmgr Works With mod_jserv

mod_oprocmgr provides infrastructure capabilities, such as automatic starting of processes, death detection and restart, and load balancing. These capabilities are enabled by a new mode, auto, for the ApJServManual directive.

Based on the configuration information provided by mod_jserv, mod_oprocmgr starts the specified number of JServ processes, managing them for the life of the servers.

Benefits of Using mod_oprocmgr With mod_jserv

mod_oprocmgr enhances the functionality and administration of JServ in several ways:

Process Management

With `ApJServManual off', only one JServ engine can be started and managed automatically. Additional servlet engines have to be manually started, monitored and stopped.

With `ApJServManual auto', any number of JServ engines can be started automatically. The process manager will continually monitor the health of these processes and kill and restart them, if necessary. You can still start JServ processes manually, if you need to.

Configuration

Configuring multiple JServ processes with `ApJServManual on' / `ApJServManual off' is more complicated and error prone. For example, a 10 process “balance” configuration requires 32 directives and 10 jserv.properties files.

Configuring multiple JServ processes with the new `auto' mode requires much less effort. For example, a 10 process “balance” configuration requires only 3 directives.

Configuring mod_jserv for Process Management

If you are already familiar with the configuration directives for mod_jserv, the configuration process for mod_oprocmgr is straightforward. The configuration files are listed below.

Changes to httpd.conf

To use mod_oprocmgr, ensure that the directives below are included in the file:

`ORACLE_HOME/Apache/Apache/conf/httpd.conf` (UNIX)

`ORACLE_HOME\Apache\Apache\conf\httpd.conf` (Windows)

```
<IfModule mod_oprocmgr.c>
  ProcNode my-sun.us.oracle.com 7777
  <IfDefine SSL>
    ProcNode my-sun.us.oracle.com 80
  </IfDefine>
  <Location /oprocmgr-service>
    SetHandler oprocmgr-service
  </Location>
 </IfModule>
```

In addition, you must specify at least one non-SSL port. For a secure website (that is, one that only accepts SSL connections), you must provide an extra non- SSL port. To do this, add the directives shown below, substituting port and address values:

```
Listen <port>
<VirtualHost _default_:port>
   SSLEngine Off
  <Location />
    order deny, allow
    deny from all
    allow from <IP address 1 of local node>
    allow from <IP address 2 of local node>
    allow from <IP address 3 of local node>
  </Location>
</VirtualHost>
```

In the LoadModule section, ensure that mod_oprocmgr is loaded after mod_osso. The call back function of mod_oprocmgr in the ’check usrid’ stage must be invoked before that of mod_osso.

Changes to jserv.properties

In the file:

ORACLE_HOME/`Apache/Jserv/etc/jserv.properties` (UNIX)

ORACLE_HOME`\Apache\Jserv\etc\jserv.properties` (Windows)

you specify the ports to which JServ will bind, as shown in the example below.

```
port=8007
```

If no ports are specified, the JServ processes will choose their ports. If you want the JServ processes to choose their ports, enter the port directive as shown below. If you eliminate the directive entirely, an error will occur.

```
port=
```

You can specify multiple ports, and separate the values with commas as shown in the example below. Note that a range of ports (9000-9010) is a valid value.

```
port=8007,9000-9010,8010
```

Changes to jserv.conf

To use mod_oprocmgr with mod_jserv, you must change the directives as indicated below in the JServ configuration file:

ORACLE_HOME/`Apache/Jserv/etc/jserv.conf` (UNIX)

ORACLE_HOME`\Apache\Jserv\conf\jserv.conf` (Windows)

ApJServManual This directive accepts a new mode, auto, which invokes the new infrastructure functionality (in which mod_oprocmgr manages processes). The syntax is:

```
ApJServManual auto
```

You can set the mode to on or off to use the standard JServ functionality.

ApJServGroup This directive defines groups for the process manager to manage for mod_jserv. If you have worked with mod_jserv, you will note that this directive replaces the ApJServBalance, ApJServHost, ApJServRoute and ApJServShmFile directives.

All JServ processes to be managed must belong to a group, and each group has its own ApJServGroup directive. If you only have one JServ process, you must define a group with just that process in it. The processes in a group are identical except for their listening ports, so requests directed to the group are distributed evenly among the processes.

The ApJServGroup directive takes four arguments: groupname, number of processes, node weight, and properties file. In the example below, the groupname is mygroup, the number of processes is 1, the node weight is 1, and the full path of the properties file used to start the JServ processes is

```
ORACLE_HOME/Apache/JServ/etc/jserv.properties

ApJServGroup mygroup 1 1 /private2/up_1022/Apache/Jserv/etc/jserv.properties
```

ApJServGroupMount This directive defines a mount point and maps it to a process group and zone. In the example below, the mount point is /servlets, the group is mygroup, and the zone is root. Note that the balance protocol is in use for routing, as in the standard JServ configuration.

```
ApJServGroupMount /servlets balance://mygroup/root
```

Place this directive after the ApJServGroup directive in the configuration file.

ApJServGroupSecretKey This directive specifies the secret key that JServ needs to authenticate clients. It can be disabled, as shown below:

```
ApJServGroupSecretKey disabled
```

When activated, the directive takes one or two arguments. In the example below, with group and filename arguments, the filename mysecretkey applies to the group mygroup.

```
ApJServGroupSecretKey mygroup /usr/local/apache/jserv/mysecretkey
```

You can supply only the filename argument, as shown below. No group is named, so the secret key filename applies to all groups.

```
ApJServGroupSecretKey /usr/local/apache/jserv/mysecretkey
```

You cannot combine directives using the one-argument syntax with directives using the two-argument syntax. If you use the two-argument syntax, the default for groups without a group-specific secret key is ’disabled’.

Place this directive after the ApJServGroup directive in the configuration file.

Warning: The secret in the secret key file specified in ApJServSecretKey must be the same as that specified by the security.secretKey directive in the jserv.properties file. If the secrets are not the same, the death detection mechanism assumes that all the servlet engine processes are dead, eliminates them, and starts new processes to replace them (repeating the cycle endlessly).

---

mod_oradav

mod_oradav is the Oracle module (an OCI application written in C) that is an extended implementation of mod_dav, and is integrated with the Oracle HTTP Server. mod_oradav can read and write to local files or to an Oracle database. The Oracle database must have an OraDAV driver (a stored procedure package) that mod_oradav calls to map WebDAV activity to database activity. Essentially, mod_oradav enables WebDAV clients to connect to an Oracle database, read and write content, and query and lock documents in various schemas.

You can configure mod_oradav to an Oracle database using standard Oracle HTTP Server directives. mod_oradav can immediately leverage other module code (such as mime_magic) in order to perform content management tasks. Most OraDAV processing activity involves streaming content to and from a content provider; and mod_oradav uses OCI streaming logic directly within the Oracle HTTP Server.

To configure mod_oradav, you enter parameters within a Location directive in httpd.conf. The Location directive specifies the DAV-enabled URL. The `DAV` keyword is followed by a single value: `On,` which tells mod_dav is to use the local file system for content.

The following example specifies that the directory `myfiles` under the Web server documents directory (`htdocs` by default) is to be DAV-enabled, along with all directories under `myfiles` in the hierarchy. (Note that there must not be any symlinks defined on `myfiles` or any of its subdirectories.)

```
<Location /myfiles>
   DAV On
</Location>
```

See Also: Chapter 7, "Configuring and Using mod_oradav"

See Also: *Oracle9iAS Portal Configuration Guide*

For information about using mod_oradav to access database schemas for access by third-party tools (such as Adobe GoLive and Macromedia Dreamweaver) and Oracle *inter*Media, see the OraDAV information available on the Oracle Technology Network at

http://otn.oracle.com.

mod_ossl

This Oracle module enables strong cryptography for the HTTP Server.

See Also: *Oracle9i Application Server Security Guide*.

mod_osso

This Oracle module enables single-sign on for the Oracle HTTP Server. mod_osso examines incoming requests and determines whether the resource requested is protected, and if so, retrieves the HTTP Server cookie for the user.

See Also: *Oracle9i Application Server Security Guide*.

Exposing the Basic Authentication URL to mod_osso

To operate an SSO server in SSL mode, you must specify one or both of these parameters in the `mod_osso.conf` file:

```
nonssl_sso_port=port
nonssl_sso_host=alternative sso server name
```

The port parameter is needed to facilitate internal communication between an Oracle HTTP Server and an SSL-enabled SSO server. At installation time, the SSORegistrar tool sets the `nonssl_sso_port` parameter as follows:

```
nonssl_sso_port=5000
```

Add the following to the httpd.conf file to expose the Basic Authentication URL to mod_osso:

```
#Use the following configuration to protect SSO Server URLs:

# SSO Server Login URL
<IfDefine SSL>
  <Location /pls/orasso/orasso.wwsso_app_admin.ls_login>
    SSLRequireSSL
  </Location>
</IfDefine>

# Change password URL
<IfDefine SSL>
  <Location /pls/orasso/orasso.wwsso_app_user.mgr.change_password>
    SSLRequireSSL
  </Location>
</IfDefine>

# External Application Login URL
<IfDefine SSL>
  <Location /pls/orasso/orasso.wwsso_app_user.mgr.change_password>
    SSLRequireSSL
  </Location>
```

mod_perl

This module embeds the Perl interpreter into the Oracle HTTP Server. This eliminates start-up overhead and enables you to write modules in Perl.

A demonstration of Perl capabilities is available from the Oracle9*i*AS Welcome page. Click the Demonstrations tab, then the J2EE and Web Cache link.

Warning: The demonstration script for this module that is shipped with Oracle9*i*AS should be disabled in production environments. It is included only to verify that the installation was successful.

See Also: *mod_perl Guide*

Database Usage Notes

This section provides information for mod_perl users working with databases. It explains how to test a local database connection and set character forms.

Testing the Database Connection

Below is a sample Perl script for testing the database connection of a local seed database. To use the script to test another database connection, you must replace `scott/tiger` with the user name and password for the target database.

```
##### Perl script start ######
use DBI;
print "Content-type: text/plain\n\n";
$dbh = DBI->connect("dbi:Oracle:", "scott/tiger", "") || die $DBI::errstr;
 $stmt = $dbh->prepare("select * from emp order by empno")|| die $DBI::errstr;
$rc = $stmt->execute() || die $DBI::errstr;
while (($empno, $name) = $stmt->fetchrow()) { print "$empno $name\n"; }
warn $DBI::errstr if $DBI::err;
die "fetch error: " . $DBI::errstr if $DBI::err;
$stmt->finish() || die "can't close cursor";
$dbh->disconnect() || die "cant't log off Oracle";
##### Perl script End ######
```

Using SQL NCHAR Datatypes

SQL NCHAR datatypes have been refined in Oracle9*i*, and are now called reliable Unicode datatypes. SQL NCHAR datatypes such as NCHAR, NVARCHAR2 and NCLOB allow you to store any Unicode characters regardless of the database character set. The character set for those datatypes is specified by the national character set, which is either AL16UTF-16 or UTF8. See the Oracle9*i* documentation for more about SQL NCHAR datatypes.

This release of DBD::Oracle supports SQL NCHAR datatypes and provides driver extension functions to specify the character form for data binding. The following script shows an example to access SQL NCHAR data:

```
# declare to use the constants for character forms
use DBD::Oracle qw(:ora_forms);
# connect to the database and get the database handle
$dbh = DBI->connect( ... );
# prepare the statement and get the statement handle
$sth = $dbh->prepare( 'SELECT * FROM TABLE_N WHERE NCOL1 = :nchar1' );
# bind the parameter of a NCHAR type
$sth->bind_param( ':nchar1', $param_1 );
# set the character form to NCHAR
```

```
$sth->func( { ':nchar1' => ORA_NCHAR } , 'set_form' );
$sth->execute;
```

As shown above, the set_form function is provided as a private function that you can invoke with the standard DBI func() method. It takes an anonymous hash that specifies which placeholder should be associated with which character form. The valid values of character form are either ORA_IMPLICIT or ORA_NCHAR. Setting the character form to ORA_IMPLICIT causes the application's bound data to be converted to the database character set, and ORA_NCHAR to the national character set. The default form is ORA_IMPLICIT.

Another function is provided to specify the default character set form as follows:

```
# specify the default form to be NCHAR
$dbh->func( ORA_NCHAR, 'set_default_form' );
```

After this call is made, the form of all parameters is ORA_NCHAR, unless otherwise specified with set_form calls. Note that unlike the set_form function, this is a function on the database handle, so every statement from the database handle with its default form specified will have the form of your choice by default.

set_form

This function sets the character form for parameter(s). Valid forms are either ORA_IMPLICIT(default) or ORA_NCHAR. The constants are available as: `ora_forms` in DBD::Oracle.

Examples:

```
# a declaration example for the constants ORA_IMPLICIT and ORA_NCHAR
use DBD::Oracle qw(:ora_forms);
# set the character form for the placeholder :nchar1 to NCHAR

$sth->func( { ':nchar1' => ORA_NCHAR } , 'set_form' );

# set the character form using the positional index
$sth->func( { 2 => ORA_NCHAR } , 'set_form' );
# set the character form for multiple placeholders at once
$sth->func( { 1 => ORA_NCHAR, 2 => ORA_NCHAR } , 'set_form' );
```

set_default_form

This function sets the default character form for a database handle.

Example:

```
$dbh->func( ORA_NCHAR , 'set_default_form' );
```

mod_plsql

This Oracle module connects the Oracle HTTP Server to mod_plsql, enabling you to create Web applications using Oracle stored procedures. This section contains the following topics:

mod_plsql Configuration Files

DAD Parameters

Sample DADs

Configuring mod_plsql For Use With WebDB 2.x

mod_plsql Configuration Files

The mod_plsql configuration files are related in a “configuration tree”, which is implemented as shown below.

The primary Oracle HTTP Server configuration file

```
ORACLE_HOME/Apache/Apache/conf/httpd.conf
```

contains an include directive for:

```
ORACLE_HOME/Apache/Apache/conf/oracle_apache.conf
```

`oracle_apache.conf` contains an include directive for:

```
ORACLE_HOME/Apache/modplsql/conf/plsql.conf
```

`plsql.conf` contains an include directive for:

```
ORACLE_HOME/Apache/modplsql/conf/dads.conf
ORACLE_HOME/Apache/modplsql/conf/cache.conf
```

plsql.conf

This file contains the LoadModule directive to load mod_plsql into the Oracle HTTP Server, global settings for mod_plsql, and include directives for dads.conf and cache.conf. An example is shown below.

```
###############################################################################
#                        mod_plsql Configuration File                          #
###############################################################################
#
#In a default install, this file gets included as per the following tree
#   httpd.conf (under $ORACLE_HOME/Apache/Apache/conf)
#     |
#     |--> oracle_apache.conf (under $ORACLE_HOME/Apache/Apache/conf)
#            |
#            |
#            |--> plsql.conf (under $ORACLE_HOME/Apache/modplsql/conf)
#                   |
#                   |-----> dads.conf (under $ORACLE_HOME/Apache/modplsql/conf)
#                   |-----> cache.conf (under $ORACLE_HOME/Apache/modplsql/conf)
#
# Tell Apache to load the Modplsql module
LoadModule plsql_module %ORACLE_HOME%/Apache/modplsql/bin/modplsql.%SO_EXT%

# Load in the setting only if plsql_module is loaded
<IfModule mod_plsql.c>

# Global Settings Section
# This section contains modplsql directives that applies to all DADs.

# Log mode of modplsql.
# To view more details about the internal processing of modplsql, please set
# this directive to 'On'. This will cause modplsql to start logging for every
# request that is processed. The log files will be generated specified by
# the PlsqlLogDirectory directive (defined below).
#
# Logging is meant to be used for debugging purposes only. When logging is
# enabled, there will be a lot of log files generated under
# $ORACLE_HOME/Apache/modplsql/logs (or as configured by the directive
# PlsqlLogDirectory). This parameter should be set to 'Off' unless recommended
# by Oracle support to debug problems with mod_plsql.
#
# Syntax: PlsqlLogEnable [Off/On]
# Default: Off
PlsqlLogEnable Off
```

```
# Log directory of modplsql.
# Set the directory name of the place where log files should be generated when
# logging is enabled. To avoid possible confusion about the location of this
# directory, an absolute path is recommended
#
# On Unix, this directory must have write permissions by the owner of the
# child httpd processes. In other words, if Apache is running as user nobody,
# then this directory must have its permissions set so that user nobody can
# write to it.
#
# Syntax: PlsqlLogDirectory [directory]
# Default: [none]
PlsqlLogDirectory %ORACLE_HOME%/Apache/modplsql/logs

# DMS logging of modplsql.
# This turns on/off the DMS logging for modplsql. Usually, this is turned on
# in order to monitor modplsql.
#
# If you do not plan to monitor the performance of your site using the OEM
# Tool, then turning this parameter off will give you a small benefit in
# performance.
#
# Syntax: PlsqlDMSEnable [On/Off]
# Default: On
# PlsqlDMSEnable On


# Cache Settings Section
# Load in the cache settings by including it here
include %ORACLE_HOME%/Apache/modplsql/conf/cache.conf

# Data Access Descriptors Settings Section
# Load in the DADs settings by including it here
include %ORACLE_HOME%/Apache/modplsql/conf/dads.conf

</IfModule>
```

dads.conf

This file contains the configuration parameters for the PL/SQL Database Access Descriptor (DAD).

A DAD is a set of values that specify how mod_plsql connects to a database server to fulfill an HTTP request. Besides the connection details, a DAD contains important configuration parameters for various operations in the database and for mod_plsql in general. Any web-enabled PL/SQL application which makes use of the OWA Web ToolKit must create a DAD to invoke the application.

Some typical PL/SQL applications that require DADs are:

- Oracle Portal
- Single Sign-On server
- Any OAS PL/SQL Cartridge application

DAD Format

A DAD has the format shown below. It uses the Oracle HTTP Server `<Location>` container directive to mount virtual paths to a particular DAD.

```
<Location /pls/orasso>
  SetHandler pls_handler
  Order deny,allow
  ...
</Location>
```

The `<Location>` container specifies the virtual path (the DAD path, which in the example above is `/pls/orasso`) for the Oracle HTTP Server. The directives in the container specify how requests routed to that location are to be processed. This is a mandatory parameter for a DAD.

For most PL/SQL applications, the DAD path can be any string.

For example:

```
/myapp
/plsqlapp
/cartx/owa
```

For Oracle Portal, all DADs must be prefixed with '/pls'. For example:

```
/pls/portal
/pls/portal309
/pls/portal306
```

The SetHandler directive tells the Oracle HTTP Server to forward the request to mod_plsql to handle. This is a mandatory parameter for a DAD.

Below is an example of a typical PLSQL application DAD:

```
<Location /pls/plsqlapp>
  SetHandler pls_handler
  Order deny,allow
  AllowOverride None
  PlsqlDatabaseUsername         scott
  PlsqlDatabasePassword         tiger
  PlsqlDatabaseConnectString    host:port:sid
  PlsqlDefaultPage              scott.home
  PlsqlDocumentTablename        scott.wwdoc_document
  PlsqlDocumentPath             docs
  PlsqlDocumentProcedure        scott.wwdoc_process.process_download
  PlsqlAuthenticationMode       Basic
</Location>
```

DAD Parameters

This section describes all the Database Access Descriptor (DAD) level parameters that can be specified in the `dads.conf` file. The following directives are Oracle HTTP Server supported directives. You can use any directives typically used in `<Location>` containers, such as:

```
Order deny,allow
AllowOverride None
```

PlsqlDatabaseUserName

Specifies the username to use to log in to the database.

Syntax:

```
PlsqlDatabaseUsername string
```

Default: None.

Example:

```
PlsqlDatabaseUsername scott
```

Notes:

- This is a mandatory parameter, except for a DAD that sets `PlsqlAuthenticationMode` to `Basic` and uses dynamic authentication.
- For DADs using `SingleSignOn` authentication, this parameter is the name of the schema owner.
- For DADs using WebDB 2.x, this parameter should be omitted.
- In Oracle9*i*AS Release 1, this configuration parameter was called `username`.

PlsqlDatabasePassword

Specifies the password to use to log in to the database.

Syntax:

```
PlsqlDatabasePassword string
```

Default: None.

Example:

```
PlsqlDatabasePassword tiger
```

Notes:

- This is a mandatory parameter, except for a DAD that sets PlsqlAuthenticationMode to Basic and uses dynamic authentication.
- For DADs using Single Sign-On authentication, this parameter is the name of the schema owner.
- For DADs using WebDB 2.x, this parameter should be omitted.
- For security reasons, Oracle recommends that you use the Oracle Enterprise Manager Console to configure this parameter.
- In Oracle9*i*AS Release 1, this configuration parameter was called `password`.

PlsqlDatabaseConnectString

Specifies the connection to a remote database.

Syntax:

```
PlsqlDatabaseConnectString string
```

where string can be one of the following:

- A valid TNS alias
- HOST:PORT:SID format where HOST is the hostname running the database, PORT is the port number the TNS listener is listening on, and SID is the Oracle SID name of the database instance.

Default: None.

Example:

```
PlsqlDatabaseConnectString orcl.us.oracle.com
```

or

```
PlsqlDatabaseConnectString myhost.us.oracle.com:1521:ORCL
#
```

Notes:

- If the database is running in the same Oracle home, or the environment variable `TWO_TASK` is set (called `LOCAL` on Windows NT), this parameter is unnecessary.
- If the database is running in a separate Oracle home, then this parameter is mandatory.
- If you have problems connecting to the database:
  - Check the username and password information in the DAD.
  - Make sure that you can execute commands such as:

    ```
    sqlplus DADUsername/DADPassword@string
    ```

  - Ensure that TNS_ADMIN is configured properly.
  - Verify that the HOST:PORT:SID format makes the connection go through.
  - Ensure that the TNS listener and database are running.
  - Ensure that you can ping the host from a different machine.

- In Oracle9*i*AS Release 1, this configuration parameter was called `connect_string`.

PlsqlAuthenticationMode

Specifies the authentication mode to use for allow access through this DAD.

Syntax:

```
PlsqlAuthenticationMode Basic/SingleSignOn/GlobalOwa/CustomOwa/PerPackageOwa
```

Default: Basic

Example:

```
PlsqlAuthenticationMode Basic
```

Notes:

- For Oracle Portal, you must set this parameter to `SingleSignOn`.
- For WebDB 2.x applications, this parameter must be set to `Basic`.
- Older releases of Oracle applications use the `GlobalOwa` mode.
- Custom Authentication modes (`GlobalOwa`, `CustomOwa`, `PerPackageOwa`) are used by very few PL/SQL applications.
- If the DAD is not using the `Basic` authentication, then you must include a valid username/password in the DAD configuration. For the `Basic` mode, if you wish to perform dynamic authentication, you can omit this parameter.
- In Oracle9*i*AS Release 1, this configuration parameter was derived from a combination of `enablesso` and `custom_auth`.
  - `enablesso = Yes` translates to `PlsqlAuthenticationMode SingleSignOn`
  - `custom_auth = Global` translates to `PlsqlAuthenticationMode GlobalOwa`
  - `custom_auth = Custom` translates to `PlsqlAuthenticationMode CustomOwa`
  - `custom_auth = PerPackage` translates to `PlsqlAuthenticationMode PerPackageOwa`
  - All other combinations translate to `Basic`.

PlsqlSessionCookieName

Specifies the cookie name for the Oracle Portal session.

Syntax:

```
PlsqlSessionCookieName string
```

Default: DAD name

Example:

```
PlsqlSessionCookieName portal
```

Notes:

- For DADs not using `SingleSignOn` authentication, this parameter can be omitted. In most other cases, the session cookie name should be omitted (and this parameter automatically defaults to the DAD name).
- A session cookie name must be specified only for Oracle Portal instances that need to participate in a distributed Oracle Portal environment. For those Oracle Portal nodes you want to seamlessly participate as a federated cluster, ensure that the session cookie name for all of the participating nodes is the same.
- Independent Oracle Portal nodes need to use distinct session cookie names.
- In Oracle9*i*AS Release 1, this configuration parameter was called `sncookiename`.

PlsqlSessionStateManagement

Specifies how package and session state should be cleaned up at the end of each mod_plsql request.

Setting this parameter to `StatelessWithResetPackageState` causes mod_plsql to call `dbms_session.reset_package_state` at the end of each mod_plsql request.

Setting this parameter to `StatelessWithPreservePackageState` causes mod_plsql to call `htp.init` at the end of each mod_plsql request. This cleans up the state of session variables in the OWA Web ToolKit. The PL/SQL application is responsible for cleaning up its own session state. Failure to do so will cause erratic behavior, in which a request will start recognizing or manipulating state modified in previous requests.

Setting this parameter to `StatelessWithFastResetPackageState` causes mod_plsql to call `dbms_session.modify_package_state(dbms_session.reinitialize)` at the end of each mod_plsql request. This API is a lot faster than the mode of `StatelessWithResetPackageState` and avoids some latch contention issues, but exists only in database versions 8.1.7.2 and above. This mode uses up slightly more memory than the default mode. Check the status of bug 2096244 before using this mode.

Syntax:

```
PlsqlSessionStateManagement
StatelessWithResetPackageState/StatelessWithFastResetPackageState/StatelessWithP
reservePackageState
```

Default: `StatelessWithResetPackageState`

Example:

```
PlsqlSessionStateManagement StatelessWithResetPackageState
```

Notes:

- `StatelessWithPreservePackageState` mode is only used with older releases of Oracle applications, in which Oracle9*i*AS Portal is not used.
- In Oracle9*i*AS Release 1, this configuration parameter was called `stateful`.
- An older value of `stateful=no` or `stateful=STATELESS_RESET` corresponds to `PlsqlSessionStateManagement StatelessWithResetPackageState`
- An older value of `stateful=STATELESS_FAST_RESET` corresponds to `PlsqlSessionStateManagement StatelessWithFastResetPackageState`
- An older value of `stateful=STATELESS_PRESERVE` corresponds to `PlsqlSessionStateManagement StatelessWithPreservePackageState`
- mod_plsql does not support stateful mode of operation. To equip PL/SQL applications with stateful behavior, save state in cookies and/or in the database.

PlsqlMaxRequestsPerSession

Specifies the maximum number of requests a pooled database connection should service before it is closed and re-opened.

Syntax:

```
PlsqlMaxRequestsPerSession number
```

Default: 1000

Example:

```
PlsqlMaxRequestsPerSession 1000
```

Notes:

- This parameter helps mitigate memory and resource problems that may occur during prolonged session use by a PL/SQL application.
- This parameter should not need to be changed; the default is sufficient for most cases.
- Setting this parameter to a low number can degrade performance. A case for a lower value might be an infrequently used DAD whose performance is not a concern, and for which limiting the number of requests provides some benefit. If you set this parameter very low (e.g., 1) for a DAD that is accessed frequently, the behavior described in Bug #1575624 can occur.
- In Oracle9*i*AS Release 1, the equivalent to this parameter is `reuse`. Instead of taking a value of `Yes/No`, the new parameter allows for finer control over the connection pool reuse in mod_plsql.

PlsqlDefaultPage

Specifies the default procedure to call if none is specified in the URL.

Syntax:

```
PlsqlDefaultPage string
```

Default: None.

Example:

```
PlsqlDefaultPage myschema.mypackage.home
```

Notes:

- In Oracle9*i*AS Release 1, this parameter was called `default_page`.

PlsqlDocumentTablename

Specifies the table in the database to which all documents will be uploaded.

Syntax:

```
PlsqlDocumentTablename string
```

Default: None.

Example:

```
PlsqlDocumentTablename myschema.document_table
```

For Portal:

```
PlsqlDocumentTablename portal.wwdoc_document
```

For WebDB:

```
PlsqlDocumentTablename webdb.wwv_document
```

Notes:

- For applications that do not do document uploads or downloads, this parameter may be omitted.
- Refer to the *mod_plsql User's Guide* for more details about upload and download processes and the structure of the restrictions on the document table format.
- In Oracle9*i*AS Release 1, this parameter was called `document_table`.

PlsqlDocumentPath

Specifies the access path to download a document. This is a virtual path that initiates document download from the document table. For example, if this parameter is set to `docs`, then the following URLs will start the document downloading process for URLs of the format:

```
/pls/dad/docs
/pls/plsqlapp/docs
```

Syntax:

```
PlsqlDocumentPath string
```

Default: `docs`

Example:

```
PlsqlDocumentPath docs
```

Notes:

- For applications that do not do document uploads or downloads, this parameter may be omitted.
- Refer to the *mod_plsql User's Guide* for more details about upload and download processes and the structure of the restrictions on the document table format.
- In Oracle9*i*AS Release 1, this parameter was called `document_path`.

PlsqlDocumentProcedure

Specifies the procedure to call when a document download is initiated. This procedure is called to process the download.

Syntax:

```
PlsqlDocumentProcedure string
```

Default: None.

Example:

```
PlsqlDocumentProcedure portal.wwdoc_process.process_download
```

Notes:

- For applications that do not do document uploads or downloads, this parameter may be omitted.
- Refer to the *mod_plsql User's Guide* for more details about upload and download processes and the structure of the restrictions on the document table format.
- In Oracle9*i*AS Release 1, this parameter was called `document_proc`.

PlsqlUploadAsLongRaw

Specifies the extensions to be uploaded as LONGRAW data type (as opposed to using the default BLOB data type). The default can be overridden by specifying multiline directives of file extensions for field. A value of '*' in this field will cause all documents to be uploaded as LONGRAW.

Syntax:

```
PlsqlUploadAsLongRaw string multiline
```

Default: None.

Example:

```
PlsqlUploadAsLongRaw jpg
PlsqlUploadAsLongRaw gif
```

For WebDB:

```
PlsqlUploadAsLongRaw *
```

Notes:

- For applications that do not do document uploads or downloads, this parameter may be omitted.
- Refer to the *mod_plsql User's Guide* for more details about upload and download processes and the structure of the restrictions on the document table format.
- In Oracle9*i*AS Release 1, this parameter was called `upload_as_log_raw`.

PlsqlPathAlias

Specifies a virtual path alias to map to a procedure call. This is application specific; for example, Oracle Portal sets this to `url` which means that all URLs of type `/pls/myapp/url/*` are handled as special URLs and directed to a specific procedure as configured by PlsqlPathAliasProcedure.

Syntax:

```
PlsqlPathAlias string
```

Default: None.

Example:

```
PlsqlPathAlias url
```

Notes:

- For applications that do not use path aliasing,this parameter may be omitted. Refer to the *mod_plsql User's Guide* for more details about path aliasing functionality.
- In Oracle9*i*AS Release 1, this parameter was called `pathalias`.

PlsqlPathAliasProcedure

Specifies the procedure to call when the virtual path in the URL matches the path alias as configured by `PlsqlPathAlias`.

Syntax:

```
PlsqlPathAliasProcedure string
```

Default: None.

Example:

```
PlsqlPathAliasProcedure portal.wwpth_api_alias.process_download
```

Notes:

- For applications that do not use path aliasing,this parameter may be omitted. Refer to the *mod_plsql User's Guide* for more details about path aliasing functionality.
- In Oracle9*i*AS Release 1, this parameter was called `pathaliasproc`.

PlsqlExclusionList

Specifies a pattern for excluding certain procedures, packages, or schema names from being directly executed from a browser. This is a multi-line directive in which each pattern occupies one line. The pattern is case-insensitive and can accept simple wildcards such as *, ? and [a-z]. The default patterns excluded from direct URL access are: `sys.*`, `dbms_*`, `utl_*`, `owa_*`, `owa.*`, `htp.*`, `htf.*`.

Setting this directive to `#NONE#` will disable all protection. This is not recommended for a live site, however, it is sometimes used for debugging purposes.

If this parameter is overridden, the defaults are no longer in effect. In that case, you must explicitly add the default list to the list of excluded patterns.

Syntax:

```
PlsqlExclusionList string multiline/#NONE#
```

Default:

```
sys.*
dbms_*
utl_*
owa_*
owa.*
htp.*
htf.*
```

Example:

```
PlsqlExclusionList sys.*
PlsqlExclusionList dbms_*
PlsqlExclusionList utl_*
PlsqlExclusionList owa_*
PlsqlExclusionList owa.*
PlsqlExclusionList htp.*
PlsqlExclusionList htf.*
PlsqlExclusionList myschema.private.*
```

will exclude access to URLs containing: `sys.*`, `dbms_*`, `utl_*`, `owa_*`, `owa.*`, `htp.*`, `htf.*`, `myschema.private.*`

However,

```
PlsqlExclusionList myschema.private.*
```

will only exclude access to URLs containing myschema.private.*. The system defaults will no longer be protected (this is normally done for backward compatibility only).

Notes:

- In addition to URL patterns specified with this parameter, mod_plsql also excludes any URLs containing special characters such as tabs, newlines, carriage returns, single quotes or the reverse slash. This cannot be changed.
- To add a pattern to the defaults, you must specify the default list with the pattern you have added (as in the example above).
- In Oracle9*i*AS Release 1, this parameter was called `exclusion_list`.

PlsqlCGIEnvrionmentList

Specifies overrides and or additions of CGI environment variables to the default set of environment variables passed down to a PL/SQL procedure. This is a multi-line directive of name-value pairs to be added, overridden or removed.

You can add CGI environment variables from the Oracle HTTP Server environment by specifying the variable name. To remove a CGI environment variable, set it equal to nothing. To add your own name-value pair, use the syntax `myname=myvalue`

Syntax:

```
PlsqlCGIEnvironmentList string multiline
```

Default: None.

Example:

To add and/or override:

```
PlsqlCGIEnvironmentList MYENVAR1=myvalue
```

To remove:

```
PlsqlCGIEnvironmentList MYENVAR2=
```

To add and/or override from the Oracle HTTP Server environment:

```
PlsqlCGIEnvironmentList DOCUMENT_ROOT
```

Notes:

- Environment variables added here are available in the PL/SQL application via the function `owa_util.get_cgi_env`.
- In Oracle9*i*AS Release 1, this parameter was called `cgi_env_list`.

PlsqlCompatibilityMode

Specifies the compatibility mode for running mod_plsql. If you are running mod_plsql against an pre-9.0.2 version of mod_plsql, you must set this value to 1.

Syntax:

```
PlsqlCompatibilityMode BitFlag
```

Default: 0

Example:

```
PlsqlCompatibilityMode 1
```

Notes:

- This parameter enables an old bug in mod_plsql in which mod_plsql incorrectly converted the plus symbol (+) to space characters for document downloads. Enabling the first bit in this flag will make it impossible to download documents that have a plus symbol (+) in the document name.

PlsqlNLSLanguage

Specifies the NLS_LANG variable for this DAD. This parameter overrides the NLS_ LANG environment variable. When this parameter is set, the PL/SQL Gateway uses the specified NLS_LANG to connect to the database. Once connected, an alter session command is issued to switch to the specified language and territory.

`Syntax: PlsqlNLSLanguage` *`string`*

Default: None.

Example:

```
PlsqlNLSLanguage America_American.UTF8
```

Notes:

- Most applications have `PlsqlTransferMode` set to `CHAR` which means that the character set in `PlsqlNLSLanguage` needs to match the character set of the database. In one special case, where the database and mod_plsql are both using fixed-size character sets, and the character set width matches, the character set can be different. The response character set is always the mod_plsql character set.
- `If PlsqlTransferMode` is set to `RAW`, then this parameter can be ignored.
- In Oracle9*i*AS Release 1, this parameter was called `nls_lang`.

PlsqlFetchBufferSize

Specifies the number of rows of content to fetch from the database for each trip (using either `owa_util.get_page` or `owa_util.get_page_raw`). For large generated pages, setting this parameter higher can decrease the number of trips to the database to get the content. However, mod_plsql memory usage will increase.

For Japanese, Chinese or multi-byte character set languages, setting this parameter to 256 will yield better performance.

Syntax:

```
PlsqlFetchBufferSize number
```

Default: 128

Example:

```
PlsqlFetchBufferSize 256
```

Notes:

- This parameter is changed only for performance reasons. The minimum value for this parameter is 28, but it is seldom reduced.
- Change this parameter only under the following circumstances:
  - The average response page is large and you want to reduce the number of round-trips mod_plsql makes to the database to fetch the response.
  - The character set in use is multi-byte, and you want to compensate for the problem of `get_page` or `get_page_raw` fetching fewer bytes per row (calculations in the OWA Web ToolKit are character-based and in the case of multi-byte characters, OWA packages assume a worst-case character byte size and do not attempt to pack each row to its maximum).
- In Oracle9*i*AS Release 1, this parameter was called `response_array_size`.

PlsqlErrorStyle

Specifies the Error Reporting Mode for mod_plsql errors. This parameter accepts the following values:

`ApacheStyle`: This is the default mode. In this mode, mod_plsql indicates to Oracle HTTP Server the HTTP error that was encountered. The Oracle HTTP Server then generates the error page. This can be used with the Oracle HTTP Server `ErrorDocument` directive to produce customized error messages.

`ModplsqlStyle`: mod_plsql generates the error pages, usually a short message indicating the PL/SQL error that was encountered and PL/SQL exception stack, if any. For example:

```
scott.foo PROCEDURE NOT FOUND
```

`DebugStyle`: This mode provides more details than `ModplsqlStyle`. mod_plsql provides more details about the URL, parameters and also produces server configuration information. This mode is for debugging purposes only. Do not use this in a production system, since displaying internal server variables could be a security risk.

```
Syntax: PlsqlErrorStyle ApacheStyle/ModplsqlStyle/DebugStyle
```

Default: `ApacheStyle`

Example:

```
PlsqlErrorStyle ModplsqlStyle
```

Notes:

- In Oracle9*i*AS Release 1, this parameter was called `error_style`.

PlsqlTransferMode

Specifies the transfer mode for data from the database back to mod_plsql. Most applications use the default value of CHAR.

Syntax:

```
PlsqlTransferMode CHAR/RAW
```

Default: CHAR

Example:

```
PlsqlTransferMode CHAR
```

Notes:

- This parameter only needs to be changed to enable sending back responses in different character sets from the same DAD. In such a case, the CHAR mode is useless, since it will always convert the response data from the database character set to the mod_plsql character set.
- In Oracle9*i*AS Release 1, RAW transfer mode was not supported.

PlsqlBeforeProcedure

Specifies the procedure to be invoked before calling the requested procedure. This enables you to put a hook point before the requested procedure is called. This is useful in doing SQL*Traces / SQL Profiles while debugging a problem with the requested procedure. This is also useful when you want to ensure that a specific call be made before running every procedure. This is an internal parameter and can be removed without notice.

Syntax:

```
PlsqlBeforeProcedure string
```

Default: None.

Example:

```
PlsqlBeforeProcedure portal.mypkg.mybeforeproc
```

Notes:

- For all but debugging purposes, this parameter should be omitted. You could use this parameter to start SQL Trace / SQL Profiling.
- In Oracle9*i*AS Release 1, this parameter was called `before_proc`.

PlsqlAfterProcedure

Specifies the procedure to be invoked after calling the requested procedure. This enables you to put a hook point after the requested procedure is called. This is useful in doing SQL*Traces / SQL Profiles while debugging a problem with the requested procedure. This is also useful when you want to ensure that a specific call be made after running every procedure. This is an internal parameter and can be removed without notice.

Syntax:

```
PlsqlAfterProcedure string
```

Default: None.

Example:

```
PlsqlAfterProcedure portal.mypkg.myafterproc
```

Notes:

- For all but debugging purposes, this parameter should be omitted. You could use this parameter to stop SQL Trace / SQL Profiling.
- In Oracle9*i*AS Release 1, this parameter was called `after_proc`.

PlsqlBindBucketLengths

Specifies the rounding size to use while binding the number of elements in a collection bind. While executing PL/SQL statements, the Oracle database maintains a cache of PL/SQL statements in the shared SQL area, and attempts to reuse the cached statement if the same statement is executed again. Oracle's matching criteria requires that the statement texts be identical, and that the bind variable data types match. Unfortunately, the type match for strings is sensitive to the exact byte size specified, and for collection bindings is also sensitive to the number of elements in the collection. Since mod_plsql binds statements dynamically, the odds of hitting the shared cache are low, and it may fill up with near-duplicates and lead to contention for the latch on the shared area. This parameter reduces that effect by bucketing bind lengths to the nearest level.

All numbers specified should be in ascending order. After the last specified size, subsequent bucket sizes will be assumed to be twice the last one.

Syntax:

```
PlsqlBindBucketLengths number multiline
```

Default: 4,20,100,400

Example:

```
PlsqlBindBucketLengths  4
PlsqlBindBucketLengths  25
PlsqlBindBucketLengths  125
```

Notes:

- This parameter is relevant only of you are using procedures with array parameters, and passing varying number of parameters to the procedure.
- The default should be sufficient for most PL/SQL applications.
- To see if this parameter needs to be changed, check the number of versions of a SQL statement in the SQL area.
- Consider using flexible parameter passing to reduce the problem.

- In Oracle9*i*AS Release 1, this parameter was called `bind_bucket_lengths`.

PlsqlBindBucketWidths

Specifies the rounding size to use while binding the number of elements in a collection bind. While executing PL/SQL statements, the Oracle database maintains a cache of PL/SQL statements in the shared SQL area, and attempts to reuse the cached statement if the same statement is executed again. Oracle's matching criteria requires that the statement texts be identical, and that the bind variable data types match. Unfortunately, the type match for strings is sensitive to the exact byte size specified, and for collection bindings is also sensitive to the number of elements in the collection. Since mod_plsql binds statements dynamically, the odds of hitting the shared cache are low, and it may fill up with near-duplicates and lead to contention for the latch on the shared area. This parameter reduces that effect by bucketing bind widths to the nearest level.

All numbers specified should be in ascending order. After the last specified size, subsequent bucket sizes will be assumed to be twice the last one.

The last bucket width must be equal to or less than 4000. This is due to the restriction imposed by OCI where array bind widths cannot be greater than 4000.

Syntax:

```
PlsqlBindBucketWidths number multiline
```

Default:

```
32,128,1450,2048,4000
```

Example:

```
PlsqlBindBucketWidths  40
PlsqlBindBucketWidths  400
PlsqlBindBucketWidths  2000
```

Notes:

- This parameter is relevant only of you are using procedures with array parameters, and passing varying number of parameters to the procedure.
- The default should be sufficient for most PL/SQL applications.
- To see if this parameter needs to be changed, check the number of versions of a SQL statement in the SQL area.
- Consider using flexible parameter passing to reduce the problem.

- In Oracle9*i*AS Release 1, this parameter was called `bind_bucket_widths`.

PlsqlAlwaysDescribeProcedure

Specifies whether mod_plsql should describe a procedure before trying to execute it. If this is set to On, then mod_plsql will always describe a procedure before invoking it. Otherwise, mod_plsql will only describe a procedure when its internal heuristics have interpreted a parameter type incorrectly.

Syntax:

```
PlsqlAlwaysDescribeProcedure On/Off
```

Default: Off

Example:

```
PlsqlAlwaysDescribeProcedure Off
```

Notes:

- For performance reasons, you should leave this parameter set to Off, except for debugging purposes.
- In Oracle9*i*AS Release 1, this parameter was called `always_desc`.

PlsqlIdleSessionCleanupInterval

Time after which idle sessions are closed and cleaned up. This directive is used in conjunction with connection pooling of database connections and sessions in mod_plsql. When a session is not used for the specified amount of time, that session will be closed and freed. This is so that unused sessions can be cleaned up and memory be freed on the database side.

Syntax:

```
PlsqlIdleSessionCleanupInterval number
```

Default: 15 minutes

Example:

```
PlsqlIdleSessionCleanupInterval 10
```

Notes:

- For most installations, the default value should suffice.
- Setting this parameter to a low number will accelerate the cleanup of unused database sessions. However, a very low number may adversely affect the connection pooling in mod_plsql.
- If the number of open database sessions is not a concern, you might want to increase the value of this parameter for best performance. In such a case, for a site that is accessed frequently enough that sessions are never idle for the length of time specified in the session cleanup interval, you could adjust the PlsqlMaxRequestsPerSession parameter in such a way that a pooled database session is certain to be recycled on a regular basis.

Sample DADs

This section contains some examples of DADs used in Oracle9*i*AS applications.

Portal 9.0.2 DAD

```
<Location /pls/portal>
  SetHandler pls_handler
  Order deny,allow
  AllowOverride None
  PlsqlDatabaseUsername         portal
  PlsqlDatabasePassword         portal
  PlsqlDatabaseConnectString    host:port:sid
  PlsqlDefaultPage              portal.home
  PlsqlDocumentTablename        portal.wwdoc_document
  PlsqlDocumentPath             docs
  PlsqlDocumentProcedure        portal.wwdoc_process.process_download
  PlsqlAuthenticationMode       SingleSignOn
  PlsqlPathAlias                url
  PlsqlPathAliasProcedure       portal.wwpth_api_alias.process_download
  PlsqlSessionCookieName        portal
</Location>
```

Login Server instance DAD

```
<Location /pls/orasso>
  SetHandler pls_handler
  Order deny,allow
  AllowOverride None
  PlsqlDatabaseUsername         orasso
  PlsqlDatabasePassword         orasso
  PlsqlDatabaseConnectString    host:port:sid
  PlsqlDefaultPage              orasso.home
  PlsqlDocumentTablename        orasso.wwdoc_document
  PlsqlDocumentPath             docs
  PlsqlDocumentProcedure        orasso.wwdoc_process.process_download
  PlsqlAuthenticationMode       SingleSignOn
  PlsqlPathAlias                url
  PlsqlPathAliasProcedure       orasso.wwpth_api_alias.process_download
  PlsqlSessionCookieName        orasso
</Location>
```

Portal 3.0.x DAD

```
<Location /pls/portal30>
  SetHandler pls_handler
  Order deny,allow
  AllowOverride None
  PlsqlDatabaseUsername         portal30
  PlsqlDatabasePassword         portal30
  PlsqlDatabaseConnectString    host:port:sid
  PlsqlDefaultPage              portal30.home
  PlsqlDocumentTablename        portal30.wwdoc_document
  PlsqlDocumentPath             docs
  PlsqlDocumentProcedure        portal30.wwdoc_process.process_download
  PlsqlAuthenticationMode       SingleSignOn
  PlsqlPathAlias                url
  PlsqlPathAliasProcedure       portal30.wwpth_api_alias.process_download
  PlsqlSessionCookieName        portal30
  PlsqlCompatibilityMode        1
</Location>
```

WebDB DAD

```
<Location /pls/webdb>
  SetHandler pls_handler
  Order deny,allow
  AllowOverride None
  PlsqlDatabaseConnectString    host:port:sid
  PlsqlDefaultPage              webdb.home
  PlsqlDocumentTablename        webdb.wwv_document
  PlsqlDocumentPath             docs
  PlsqlUploadAsLongRaw          *
  PlsqlDocumentProcedure        webdb.wwv_document.process_download
</Location>
```

cache.conf

This file contains the cache settings for mod_plsql. An example is shown below.

```
###############################################################################
#                   Modplsql Caching Configuration File                       #
###############################################################################

# Note: this file should be appended or included into your plsql.conf

# This file specifies the charateristics of the modplsql caching system. There
# are two types of caching being used :
# - PLSQL Cache
#   PLSQL Caching is used to cache dynamically generated contents that don't
# change often. Applications using the OWA_CACHE package, such as Oracle
# Portal, use this feature to improve performance and take some load off
# the database.
#
# - Session Cookie Cache
#   Session Cookie Caching is used to cache the cookie value generated by a
# Single Sign On server for a particular session. By enabling this feature,
# a roundtrip to the database to obtain a user's credentials is avoided,
# thereby, improving performance. Only applications that use the Single Sign
# On will benefit from this feature.

# Turn caching on or off.
# For maximum performance, turn on caching. Please note that only applications
# that support PLSQL caching, such as Oracle Portal, will benefit from this
# feature.
#
# The only time caching should be turn off is during debugging and caching
# is the suspect of the problem. Otherwise, in a production environment, it
# should always be turned on.
#
# If you are absolutely sure that your application does not make use of
# Oracle Portal or the Oracle Login Server and does not in any way make use of
# the OWA_CACHE packages in the OWA ToolKit, then you could choose to disable
# caching
#
# Syntax: PlsqlCacheEnable [On/Off]
# Default: Off
PlsqlCacheEnable On

# Set directory to write the cache files.
# This directive specifies where to put the cached contents.
#
# For PLSQL cache, all cache files will be created under a directory called
# "plsql" relative to specified caching directory.
#
# For Session Cookie cache, all cache files will be created under a directory
# called "session" relative to specified caching directory.
#
# This directory must exists or Apache will not start.
#
# On Unix, this directory must have write permissions by the owner of the
# child httpd processes. In other words, if Apache is running as user nobody,
# then this directory must have its permissions set so that user nobody can
# write to it.
#
# Syntax: PlsqlCacheDirectory [directory]
# Default: [none]
PlsqlCacheDirectory %ORACLE_HOME%/Apache/modplsql/cache

# Set the total size of the cache.
# This setting limits the amount of space the cache is allowed to use. Both
# PLSQL cache and Session Cookie cache shares this cache space. Please
# note that this setting is not a hard limit. It might exceed the limit
# temporarily during normal processing. This is normal behavior.
#
# This parameter takes bytes as the value. Therefore :
# 1 Megabyte  = 1048576  bytes
# 10 Megabyte = 10485760 bytes
#
# Syntax: PlsqlCacheTotalSize [number]
# Default: 20971520 (20 Megabyte)
PlsqlCacheTotalSize 20971520
```

```
# Set the size of the space used by the cache after cleanup has been performed.
#
# Cleanup happens whenever cleanup starts and the total size of the cache is
# exceeded. This directive allows you to specifies the total size of the
# cache to maintain after the cleanup has been performed. This ensures that
# a large of the cache is still around after cleanup.
#
# This directive is useful in fine tuning the cleanup algorithm. Since cleanup
# takes quite some time to complete, this directive allows you to tune it to
# your environment. If it is set to a high number, cleanup will finish faster
# and takes less CPU cycles, but it will happen more frequently because it
# didn't thoroughly clean the cache in each run. If it is set to a low number,
# cleanup will run longer, but it will happen less frequently because it
# did thoroughly clean the cache in each run. Therefore, depending on how a
# system uses the cache system, this setting can be tweaked to best match it.\
#
# In general, setting this directive to about 50-70% of the total cache size
# will be sufficient. For example:
#
# PlsqlCacheTotalSize    1000000
# PlsqlCacheCleanupSize   600000
#
# This parameter takes bytes as the value.
#
# Syntax: PlsqlCacheCleanupSize [number]
# Default: 10485760 (10 Megabyte)
PlsqlCacheCleanupSize 10485760

# Set the amount of time (in minutes) in which the cleanup should start.
#
# This directive allows you to control when cleanup actually happens. This
# interval is amount of time passed after the previous cleanup session. In
# general, if you are low on disk space, set this number to a small amount
# of time will help free up used disk space. However, you have have lots of
# disk space, setting this to a high number will ensure better cache hits.
#
# Syntax: PlsqlCacheCleanupInterval [number]
# Default: 1440 (24 hours)
PlsqlCacheCleanupInterval 720

# Set the maximum size a cache file can be.
#
# This directive is to prevent the case in which one file can fill up the
# entire cache. In general, setting this directive to about 1-10% of the
# total cache size will be sufficient.
```

```
#
# Syntax: PlsqlCacheMaxSize [number]
# Default: 1048576 (1 Megabyte)
PlsqlCacheMaxSize 1048576
```

Configuring mod_plsql For Use With WebDB 2.x

Although WebDB 2.x is not certified with Oracle9*i*AS Release 2 (9.0.2), there are no known issues that would prevent you from using WebDB 2.x with mod_plsql. Follow these steps to configure mod_plsql to run with WebDB 2.x instances:

1. Drop any older OWA packages in OWA_PUBLIC or OAS_PUBLIC.
2. Install the latest OWA packages shipped with mod_plsql.
3. Specify the following DAD configuration in `dads.conf` for the WebDB 2.x schema:

   ```
   PlsqlAuthenticationMode Basic
   PlsqlDocumentTablename schema.wwv_document
   PlsqlUploadAsLongRaw **
   ```

4. Connect to the database as the owner of the site and run `ORACLE_HOME/Apache/modplsql/owa/wwvdocs.sql` and `ORACLE_HOME/Apache/modplsql/owa/wwvdocb.plb`.

   This enables WebDB document upload / download capability for 2.x websites.

mod_proxy

This module provides proxy capability for FTP, CONNECT (for SSL), HTTP / 0.9 and HTTP / 1.0.

> **See Also:** Module mod_proxy in the Apache Server documentation

mod_rewrite

This module provides an engine for rewriting URLs.

> **See Also:** Module mod_rewrite in the Apache Server documentation

mod_setenvif

This module enables you to set environment variables based on characteristics of a request.

See Also: Module mod_setenvif in the Apache Server documentation

mod_so

This module loads executable code and modules into the server at start-up time.

See Also: Module mod_so in the Apache Server documentation

mod_speling

This module attempts to correct misspelled or miscapitalized URLs.

See Also: Module mod_speling in the Apache Server documentation

mod_status

This module displays an HTML page of server activity and performance.

See Also: Module mod_status in the Apache Server documentation

mod_unique_id

This module creates a unique id for each request.

See Also: Module mod_unique_id in the Apache Server documentation

This module is available for UNIX systems only.

mod_userdir

This module maps requests to user-specific directories.

See Also: Module mod_userdir in the Apache Server documentation

mod_usertrack

This module tracks user activity by creating a clickstream log.

See Also: Module mod_usertrack in the Apache Server documentation

mod_vhost_alias

This module enables dynamically configured mass virtual hosting.

See Also: Module mod_vhost_alias in the Apache Server documentation

7

Configuring and Using mod_oradav

This chapter describes distributed authoring and versioning concepts, and explains how to configure and use `mod_oradav`. `mod_oradav` enables you to use OraDAV to access content in an Oracle database from a Web browser or a WebDAV client. It includes the following major sections:

- Concepts
- OraDAV Users
- Usage Model
- OraDAV Configuration Parameters
- WebDAV Security Considerations
- OraDAV Performance Considerations
- Usage Notes

Concepts

The term *OraDAV* refers to the capabilities available through the mod_oradav module. mod_oradav is an extended implementation of mod_dav, which is an implementation of the WebDAV specification. This section explains these concepts.

WebDAV

WebDAV is a protocol extension to HTTP 1.1 that supports distributed authoring and versioning. With WebDAV, the Internet becomes a transparent read and write medium, where content can be checked out, edited, and checked in to a URL address.

WebDAV enables collaboration among authors building Web sites. WebDAV also serves as universal read and write access protocol to arbitrary hierarchies of content (not necessarily Web sites). With WebDAV, you can save content to a URL provided by an Internet service provider (ISP) and then be able to access and optionally change that content from various devices.

WebDAV was initiated as an IETF standard. The first phase of WebDAV is specified in RFC 2518, which provides the basic primitives for managing hierarchies of information, locking, reading, writing, and querying properties of a WebDAV document. Subsequent work on WebDAV is ongoing and is focusing on completing issues relating to content management over the Web. This includes WebDAV authentication and authorization (access controls), versioning, bindings, ordered collections, and querying (DAV Advanced Searching and Locating).

Microsoft Web folders is a WebDAV client in Windows 2000 and on NT (using Internet Explorer 5.0). Windows 2000 applications and the IIS server support WebDAV, meaning that you can start a Microsoft Office application and specify a URL, edit the content, and save it back to the URL from which it was retrieved. WebDAV also has Java Clients (such as DAV Explorer), open source tools (such as Cadaver and Sitecopy), and Apple GUI tools (such as Goliath).

mod_dav

mod_dav is the Apache Software Foundation's native implementation of the WebDAV specification. Originally, mod_dav was a third-party Apache module; however, as of Apache 2.0, mod_dav is included.

mod_oradav

mod_oradav is the Oracle module (an OCI application written in C) that is an extended implementation of mod_dav, and is integrated with the Oracle HTTP Server. mod_oradav performs read / write activity to local files and to Oracle databases. The Oracle databases must have an OraDAV driver (a stored procedure package) that mod_oradav calls to map WebDAV activity to database activity. Essentially, mod_oradav enables mod_dav to connect to an Oracle database, read and write content, and query and lock documents in various schemas.

You can configure mod_oradav to an Oracle database using standard Oracle HTTP Server directives. mod_oradav can immediately leverage other module code (such as mime_magic) in order to perform content management tasks. Most WebDAV processing activity involves streaming content to and from a content provider; and mod_oradav uses OCI streaming logic directly within the Oracle HTTP Server.

OraDAV

OraDAV refers to the whole set of capabilities that are available through mod_oradav to Oracle9*i*AS users. Some OraDAV-specific terms include:

- **Apache OraDAV**: code in the Apache HTTP server that supports file-based DAV access and makes calls to Oracle.
- **OraDAV driver API**: the set of stored procedure calls that are used by the OraDAV driver to manage content in an Oracle database, providing support for the following WebDAV functions over the Internet: reading and writing documents, locking and unlocking documents, managing (creating, populating, deleting) hierarchies of information, retrieving properties associated with documents, and associating properties with specific documents.
- **OraDAV driver**: a stored procedure implementation of the OraDAV driver API that executes in Oracle and manages a repository.
- **OraDAV** *inter***Media driver**: a lightweight reference implementation of an OraDAV driver.

  The OraDAV interMedia driver is included with the interMedia Clipboard, which you can download from the Oracle Technology Network (`http://otn.oracle.com`) and install. This driver lets you use third-party tools to access files and database content; for example, Dreamweaver can use WebDAV as the protocol for transferring files between a local folder on your system to the remote site where your Web site is published.

Architecture

OraDAV fits into an architecture in which mod_oradav, within the Oracle HTTP Server, provides access to content in one or more schemas in one or more Oracle databases.

A simple form of the architecture is illustrated in Figure 7–1.

Figure 7–1 OraDAV Architecture

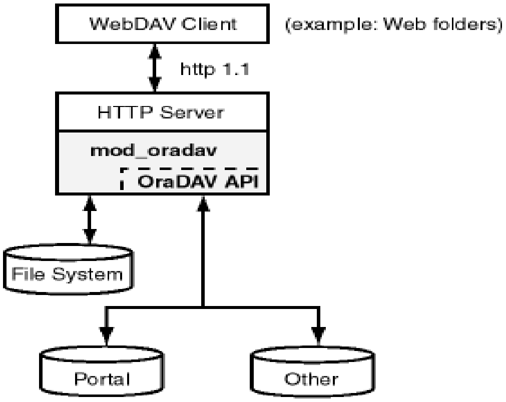


Figure 7–1 shows a WebDAV client, such as Microsoft Web folders, passing HTTP requests to the Oracle HTTP Server. If the request is for content stored in the file system (not in an Oracle database), mod_oradav handles the access. If the request is for content stored in an Oracle database, the OraDAV API handles the access.

The OraDAV API provides capabilities that are equivalent to using mod_oradav running with a file system. The following HTTP methods are supported by the OraDAV API:

- COPY
- DELETE
- MOVE
- MKCOL
- GET
- HEAD
- LOCK
- PROPFIND

- PROPPATCH
- PUT
- UNLOCK

The OraDAV API supports shared and exclusive locking, retrieving basic DAV properties, and defining and retrieving server-defined live properties or client-defined dead properties. Set-based operations such as COPY, MOVE, DELETE can be done completely by a single call to an OraDAV driver.

OraDAV Users

The primary direct users of OraDAV are Oracle HTTP Server administrators and Oracle DBAs. “End users” interact only indirectly with OraDAV through Web browsers or WebDAV client tools; they usually are not aware that they are using WebDAV technology and do not know or care about the details of the WebDAV implementation on the server.

OraDAV administration involves tasks for a webmaster and for a DBA:

- The administrator needs to know how to build, start, and stop the Oracle HTTP Server, and how to edit the `httpd.conf` file to direct URL traffic to an OraDAV driver.
- The DBA needs to know how to set up client connectivity to the Oracle database from the system running the Oracle HTTP Server, to install and administer the OraDAV driver, and perhaps to tune the content managed by the driver based on physical storage characteristics.

Usage Model

OraDAV usage can involve any combination of the following activities:

- Browsing: a read-only activity which uses WebDAV to access content in an Oracle database. Its usage model is that of a typical read-only Web site.
- Restructuring: deleting, moving, and copying content. Restructuring is usually done infrequently by a restricted set of individuals who have write access to the WebDAV content. Restructuring has the same limitations and complications that one encounters when restructuring a file directory. In some cases this directory hierarchy is owned and managed by one user. If the directory is shared, the client doing restructuring is given sole access to the hierarchy through WebDAV exclusive locks.

- Editing: modifying one or a small subset of resources in a hierarchy. Properly designed WebDAV clients will take out shared or exclusive locks on such resources to coordinate these activities.
- Property management: associating properties and attributes (for example, author) with documents for ease of lookup and for categorization. WebDAV clients assign properties to documents using the PROPPATCH directive and retrieve properties using the PROPFIND directive.

OraDAV Configuration Parameters

Configuration of OraDAV is mainly done through parameters in the `httpd.conf` file, which is used by an Oracle HTTP Server instance when it is initializing. Some configuration parameters are required for all OraDAV drivers, and others are driver-specific.

When Oracle9*i*AS is installed, all required OraDAV parameters are set with values that are designed to enable Oracle database content to be accessed through a Web browser or WebDAV client. If necessary, you can later modify the values for required parameters and specify values for optional parameters if the default values do not meet your needs. The parameters used in `httpd.conf` to support OraDAV configuration start with `DAV` and `DAVParam`. These parameters are specified within a `<Location>` directive, and they provide:

- A way of configuring how the Oracle HTTP Server connects to the database
- Coarse controls on OraDAV behavior

The `DAV` parameter indicates that a URL location is DAV-enabled. The `DAV` keyword is followed by a single value: `On` (indicating that mod_oradav is to use the local file system for content) or `Oracle` (indicating that mod_oradav is to use OraDAV for all content).

`DAVParam` parameters are used to specify name-value pairs. The required pairs are those that enable the Oracle HTTP Server to connect to an Oracle database. These include the names `OraService`, `OraUser`, and `OraPassword`.

Example 7–1 shows a configuration for accessing files on the local system. It specifies that the directory `myfiles` under the Web server documents directory (`htdocs` by default) is to be DAV-enabled, along with all directories under `myfiles` in the hierarchy. (Note that there must not be any symlinks defined on `myfiles` or any directory under it in the hierarchy.)

Example 7–1 Configuration Parameters: File System Access

```
<Location /myfiles>
   DAV On
</Location>
```

Example 7–2 shows a configuration for accessing content through an Oracle9*i*AS portal. After Portal has been installed in iAS, the Oracle HTTP Server configuration file should be populated with a <Location> directive which points to the Portal schema. In this example, the location `/portal` will be OraDAV-enabled and will (once populated with the correct values) connect to the Portal schema so that users can use WebDAV clients to access Portal data.

Example 7–2 Configuration Parameters: Portal Access

```
<Location /portal>
   DAV Oracle
   DAVParam ORACONNECT dbhost:dbport:dbsid
   DAVParam ORAUSER portal_schema
   DAVParam ORAPASSWORD portal_schema_password
   DAVParam ORAPACKAGENAME portal_schema.wwdav_api_driver
</Location>
```

Each OraDAV driver can use the `DAVParam` mechanism to create its own driver-specific settings. All `DAVParam` name-value pairs are passed to the OraDAV driver.

In addition to the OraDAV parameters, you should consider whether to specify certain DAV parameters, such as `DAVDepthInfinity`. For information about these DAV parameters, see "DAV Parameter Information" on page 7-21.

Table 7–1 lists each OraDAV parameter, whether it is required or optional, and its default value. The ORAGetSource parameter applies only to file system access; the other parameters apply only to Portal driver and other (non-file system) access.

Table 7–1 OraDAV Parameters

Name	Required?	Default Value
ORAConnect	Required[1]	(none)
ORAService	Required[1]	(none)
ORAUser	Required	(none)
ORAPassword	Required	(none)

Table 7–1 OraDAV Parameters (Cont.)

Name	Required?	Default Value
ORAPackageName	Optional	ORDSYS.DAV_ API_DRIVER
ORALockExpirationPad	Optional	0 (seconds)
ORAAllowIndexDetails	Optional	FALSE
ORAGetSource	Optional	(none)
ORACacheDirectory	Optional	(none)
ORACacheMaxResourceSize	Optional	(none)
ORACachePrunePercent	Optional	25
ORACacheTotalSize	Optional[2]	(none)

[1] Either ORAService or ORAConnect must be specified, but not both.

[2] OraCacheTotalSize is required if OraCacheDirectory is used; otherwise, do not specify the parameter.

ORAAllowIndexDetails

Applies to: Portal driver and other (non-file system) access

Required/Optional: Optional

Values: TRUE or FALSE

Default: FALSE

In an Oracle HTTP Server environment that is not OraDAV-enabled, mod_dav itself does not respond to HTTP GET requests. Instead, normal Oracle HTTP Server mechanisms are used to respond to GET requests. However, when all your content is in an Oracle database, normal Oracle HTTP Server mechanisms cannot be used to respond to GET requests, and thus OraDAV must respond to GET requests.

The ORAAllowIndexDetails parameter controls how OraDAV responds when a GET request is performed on a DAV collection and no index.html file is found in that collection (directory). In a typical Oracle HTTP Server environment, a separate module takes control, automatically generating and returning to the client HTML that represents an “index” of the resources (files) in that collection.

An OraDAV-enabled Oracle HTTP Server performs similar actions when responding to a GET request on a collection. A Description column (containing links to more detailed information about each resource) is included in the generated index when ORAAllowIndexDetails is set to TRUE.

These links consist of the URL for the resource itself followed by ?details.

The default is FALSE, in which case no “Description” column appears in the generated index, and if ?details is used in a URL, it is ignored and the URL contents are returned.

ORACacheDirectory

Applies to: Portal driver and other (non-file system) access

Required/Optional: Optional

Values: (character string)

Default: (none)

The ORACacheDirectory parameter specifies the directory to use for disk caching operations (see "Using Disk Caching with OraDAV" on page 7-17). If you do not use this parameter, disk caching is not performed for OraDAV operations.

The specified directory must exist and be readable by the Oracle HTTP Server, but cannot be visible to normal GET requests. (If the directory is visible to normal GET requests, security measures could be bypassed by users accessing the cache directory.)

The directory should not be an NFS mounted directory, because most UNIX locking mechanisms caution against this. The directory should be located on a file system that supports a "last accessed" time. For Windows systems this means using NTFS (not FAT) formatted partitions.

Do not use the cache directory for anything other than caching. Any files in the cache directory are subject to deletion.

If you use the ORACacheDirectory parameter, you must also use the ORACacheTotalSize parameter.

ORACacheMaxResourceSize

Applies to: Portal driver and other (non-file system) access

Required/Optional: Optional

Values: (integer, with optional unit character string)

Default: (none)

The ORACacheMaxResourceSize parameter specifies a maximum cachable resource size for disk caching operations (see "Using Disk Caching with OraDAV" on page 7-17). For example,

```
DAVParam ORACacheMaxResourceSize 1024KB
```

would prevent OraDAV from caching any resource larger than one megabyte. The goal is to give webmasters the ability to prevent large media files from dominating the cache. However, be aware that the performance benefit from caching a large file is greater than from caching a small file.

You can specify KB (for kilobytes) or MB (for megabytes) after an integer. If you do not specify a unit after the integer, the default unit is bytes.

ORACachePrunePercent

Applies to: Portal driver and other (non-file system) access

Required/Optional: Optional

Values: integer (1 to 100)

Default: 25

The ORACachePrunePercent parameter specifies determines a percentage of disk cache usage to be freed up when the cache is full (see "Using Disk Caching with OraDAV" on page 7-17). When the disk cache is full, the oldest files in the cache are deleted (“pruned”) until the cache disk usage is reduced by the ORACachePrunePercent value.

ORACacheTotalSize

Applies to: Portal driver and other (non-file system) access

Required/Optional: Optional, unless ORACacheDirectory is specified

Values: (integer, with optional unit character string)

Default: (none)

The ORACacheTotalSize parameter specifies the size of the cache to use for disk caching operations (see "Using Disk Caching with OraDAV" on page 7-17). Examples:

```
DAVParam ORACacheTotalSize 1GB
DAVParam ORACacheTotalSize 10485760
```

You can specify MB (for megabytes) or GB (for gigabytes) after an integer. If you do not specify a unit after the integer, the default unit is bytes. The maximum value is 4GB.

If you use the ORACacheDirectory parameter, you must also use the ORACacheTotalSize parameter.

The ORACacheTotalSize value should be large enough to hold either a significant fraction of the your Web site or all of your most frequently accessed files plus 25% or more space. If the value is too small, overall performance will degrade because of the extra work of writing BLOB data to the file system and quickly deleting files to make room for newer cache requests.

The actual space utilized by the disk cache might sometimes exceed the ORACacheTotalSize value, possibly by as much as the ORACacheMaxResourceSize value. Administrators should also be aware of file system block size issues that could cause the cache to use more disk space than the ORACacheTotalSize value.

ORAConnect

Applies to: Portal driver and other (non-file system) access

Required/Optional: Required, unless `ORAService` is specified

Values: (character string)

Default: (none)

The ORAConnect parameter specifies the Oracle database to connect to. The value must be in the following format: *`database-host:database-port:database-sid`*. For example:

```
my-pc.acme.com:1521:mysid
```

The ORAConnect parameter lets you connect to a database that is not included in the `tnsnames.ora` file.

You must specify either the `ORAService` or `ORAConnect` parameter; however, you cannot specify both.

ORAGetSource

Applies to: File system access

Required/Optional: Optional

Values: (character string in double-quotes)

Default: (none)

The `ORAGetSource` parameter applies only to file system access. It specifies one or more file extensions (including periods) to identify types of files that are not to be executed, but rather opened for editing. Use a comma to separate file extensions. For example:

```
".htm, .html, .jsp1, .jsp2"
```

The `ORAGetSource` parameter lets you open for editing files that are usually executed as a result of a GET operation.

Note: `.jsp` and `.sqljsp` files are by default opened for editing, so you do not need to specify them with the `ORAGetSource` parameter.

ORALockExpirationPad

Applies to: Portal driver and other (non-file system) access

Required/Optional: Optional

Values: (number of seconds)

Default: 0

The `ORALockExpirationPad` parameter is intended to be used in high-latency network environments, to adjust for the "refresh lock" behavior in Microsoft Office. Microsoft Office attempts to refresh locks on DAV resources just before the lock is set to expire. However, if there is network congestion between the Microsoft Office client and the DAV server, the refresh request might arrive too late, that is, after the lock has expired.

OraDAV periodically looks for locks on resources that have expired and deletes those locks. The `ORALockExpirationPad` parameter can be used to provide some additional ("pad") time between when a lock expires and when that lock is deleted. For example, if ORALockExpirationPad is set to 120, OraDAV does not actually delete locks until at least two minutes after the expiration time.

ORAPackageName

Applies to: Portal driver and other (non-file system) access

Required/Optional: Required

Values: (character string)

Default: ORDSYS.DAV_API_DRIVER

The `ORAPackageName` parameter identifies the OraDAV driver implementation that is to be called when issuing OraDAV commands. The default is the OraDAV *inter*Media driver, which is the ORDSYS.DAV_API_DRIVER package.

ORAPassword

Applies to: Portal driver and other (non-file system) access

Required/Optional: Required

Values: (character string)

Default: (none)

The ORAPassword parameter specifies the password associated with the user specified by the ORAUser parameter.

ORAService

Applies to: Portal driver and other (non-file system) access

Required/Optional: Required, unless ORAConnect is specified

Values: (character string matching an entry in the tnsnames.ora file)

Default: (none)

The ORAService parameter specifies the Oracle database to connect to. The specified value must match a SID value in the tnsnames.ora file. For example: mydbsid.mydomain.com

To connect to a database that is not included in the tnsnames.ora file, use the ORAConnect parameter.

You must specify either the ORAService or ORAConnect parameter; however, you cannot specify both.

ORAUser

Applies to: Portal driver and other (non-file system) access

Required/Optional: Required

Values: (character string)

Default: (none)

The ORAUser parameter specifies the database user (schema) to use when connecting to the service specified by the ORAService parameter.

This user must have been granted the following privileges:

- CONNECT
- RESOURCE
- CREATE TABLESPACE
- DROP TABLESPACE
- CREATE ANY TRIGGER

Other Notes

All OraDAV parameters are passed from the Oracle HTTP Server to the routines in the ORAPackageName package as part of the context parameter.

The keys are uppercase in the Oracle HTTP Server (for example, ORAUSER), but the values are not (for example, scott).

WebDAV Security Considerations

Because WebDAV enables read-write capabilities, users on the Internet can write to your Web site or to an Oracle database. A major concern is preventing users from placing an inappropriate file (a "Trojan horse") that can execute on the Web server system. If the WebDAV configuration an authorization is not set up properly, an inappropriate file from the file system can be executed. (This problem does not apply to content from an Oracle database, because such content cannot execute in the middle tier.)

The HTTP protocol issues GET requests both to static and executable files, without differentiation. The Oracle HTTP Server executes files based on their location or extension. For example, a shell script (which typically has no file extension) will be executed if it is in the cgi-bin directory, but will be retrieved as a static text file if it is in the htdocs directory. On the other hand, a Java server page, which has a .jsp extension, will normally be executed regardless of its location. However, by default, mod_oradav prevents a WebDAV-enabled directory from executing a .jsp or .sqljsp file. For a file with one of these extensions, mod_oradav reads the content directly, bypassing any Oracle HTTP Server logic that attempts to execute the file. Files with these extensions are retrieved as having the text/plain MIME type and can be edited. You can add to the list of file types that are never to be executed and always retrieved as text/plain by using the ORAGetSource parameter.

One way to limit execution of files is to use the Apache ForceType directive in a <Location> directive. This forces all content under a location to be retrieved as text/plain. However, this simple and sweeping approach may not be what you want in many cases, wherein you want the standard behavior associated with the actual MIME type (for example, for .gif files) to be used.

To decide how to handle these security issues with content on file systems, you should determine what kinds of WebDAV users are going to have access to the content. WebDAV users typically fall into two categories: Web authors who want to collaborate and manage a Web site, and end users who want to use WebDAV as a public storage area. End users should never be able to upload and execute a file, so for end users you may want to specify many file extensions with the ORAGetSource parameter or to use the ForceType directive.

Be sure to apply the standard Basic or Digest authentication and authorization mechanisms supported by the Oracle HTTP Server. You probably want to do this for the default location (dav_public) in the supplied moddav.conf file. This will restrict who can use your system for remote storage, preventing unauthorized users from filling up your disks. You should always apply Oracle HTTP Server authentication and authorization to authors of the Web site.

You should also provide both an execution context and an editing context, so that Web authors (after being properly authenticated and authorized) can edit a .jsp or other executable file and then see how it executes. To do this, create an alias for the directory associated with the execution context, and then DAV-enable the aliased location. For example, assume that you want to be able to execute a script if the URL specifies the cgi-bin directory (for example, http://www.acme.com/cgi-bin/printenv) but edit the script if the URL specifies an alias named edit-cgi-bin (for example, http://www.acme.com/cgi-bin/printenv). The following configuration file entries achieve this goal, setting up edit-cgi-bin as an editing context for content in the cgi-bin directory:

```
Alias /edit-cgi-bin /usr/local/apache/cgi-bin
<Location /edit-cgi-bin>
DAV On
ForceType text/plain
</Location>
```

OraDAV Performance Considerations

This section provides information that can help you optimize the performance of various kinds of operations. It contains the following topics:

- Using Disk Caching with OraDAV
- Bypassing Web Cache for WebDAV Activities
- Using Web Cache for Browsing Activities

Using Disk Caching with OraDAV

Oracle9*i*AS can use local file system disk caching with data that is retrieved from an Oracle database. Disk caching is designed to improve the performance of HTTP GET operations on frequently accessed database data. When data is requested from the database, it is retrieved and is also stored in a disk cache on the local file system. If a subsequent request is for the same data and if the data is still in the disk cache, Oracle9*i*AS checks to see if the data has changed in the database (by examining the `etag` value); and if the data has not changed, it is retrieved from the cache, which is more efficient than retrieving it from the database unless the files are very small.

The performance benefit from disk caching is greatest with medium-size to large files (roughly 50 KB and larger). With very small files, performance can be worse with disk caching than without. For example, if the file `myfile.dat` is requested and if the file size is only 24 bytes, the time required for copying the file from the database to the local system is very small compared to the time required for accessing the database to check if the file has changed. If disk caching is not used, there is no check of the database to see if the file has changed, and the file is copied from the database in all cases.

You can set several OraDAV parameters to control disk caching for OraDAV operations:

- ORACacheDirectory
- ORACacheTotalSize
- ORACacheMaxResourceSize
- ORACachePrunePercent

If you specify ORACacheDirectory, disk caching for OraDAV operations is enabled; and in this case you must also specify an ORACacheTotalSize value, and you can specify ORACacheMaxResourceSize and ORACachePrunePercent values. If you do not specify ORACacheDirectory, disk caching for OraDAV operations is not enabled, and the other disk cache-related parameters are not relevant.

For reference information about each parameter, see "OraDAV Configuration Parameters" on page 7-6.

Bypassing Web Cache for WebDAV Activities

Oracle9*i*AS Web Cache is a feature that enhances performance for most Web activity, which involves client read-only operations of data on the Web server system. However, Web Cache does not cache OraDAV operations (which are designed for read / write capability). Thus, for better performance, WebDAV clients can connect directly to the Oracle HTTP Server.

To bypass Web Cache for WebDAV clients, you must add a new port for listening and specify a different virtual host when using this port. Choose any port number that is not currently used by the Oracle HTTP Server and is not in a range of reserved port numbers.

Note: You *cannot* use port 7778 to bypass Web Cache for WebDAV operations. If you use port 7778, copy and move operations will return the error HTTP_BAD_GATEWAY.

For example, you could choose port number 7900 and add the following lines to the `moddav.conf` file:

```
Listen 7900
<VirtualHost _default_:7900>
  </VirtualHost>
```

WebDAV clients will connect directly to port 7900 for better performance.

Using Web Cache for Browsing Activities

If your WebDAV clients always bypass Web Cache (see "Bypassing Web Cache for WebDAV Activities" on page 7-18), you may want to tune Web Cache for read-only clients such as Web browsers. To do so, add the `DAVOraWebCacheReadOnly On` setting for an OraDAV-enabled location in the `httpd.conf` file. For example:

```
<Location /dav_public>
  DAV On
  DAVOraWebCacheReadOnly On
</Location>
```

This setting prevents WebDAV clients from using Web Cache and thus potentially retrieving stale documents for editing. (That is, the cached version of the document might not reflect edits that were recently made.) This setting, however, does allow read-only activity by browsers and other clients to use Web Cache.

Usage Notes

This section contains usage notes relating to mod_oradav. Some of the information, including most of the material relating to DAV parameters, is taken or adapted from material written by Greg Stein (`gstein@lyra.org`) and available at the following URL:

`http://www.webdav.org/mod_dav/install.html`

Mapping Containers Under the Root Location

Following are practices to avoid when mapping containers under the root location.

- Do not map the root itself. That is, do not specify `<Location />`.
- Do not map a container as a subelement in the hierarchy to another container. For example, do not specify the following two containers: `<Location /project1>` and `<Location /project1/project2>`. However, it is acceptable to specify `<Location /project1>` and `<Location /project2>`.
- Do not create any symlinks to the container or any of its subdirectories.

Globalization Support Considerations with OraDAV

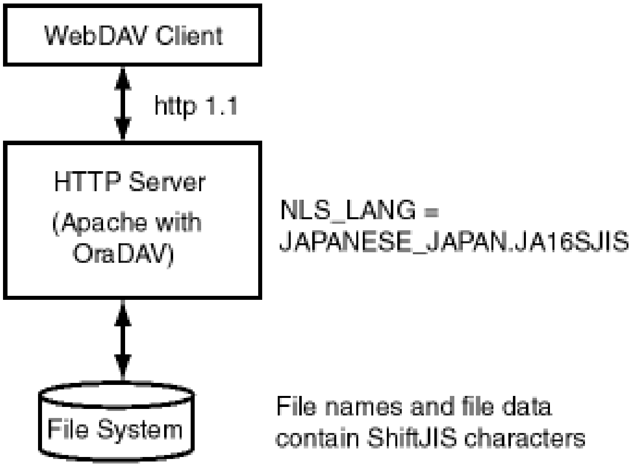
For access to database data, the character set used for client requests, such as in URLs and file names, must be compatible with the character set used for the database. Specifically, if the character set for the database is not the same as for the client requests, the character set for the database must provide for conversion of all possible characters in client requests (and thus must be a superset of the character set for client requests). That is, the character set for the database must not cause replacement characters during the conversion.

When you start the Oracle HTTP Server, the NLS_LANG environment variable must reflect the character set for client requests. For example, if file names and URLs contain Kanji characters, you can specify `NLS_LANG=JAPANESE_JAPAN.JA16SJIS` (for ShiftJIS characters). In this case, the database character set must be one that accommodates SJIS characters, for example, UTF8.

For access to the local file system (as opposed to database access), the character set for the file system must be the same as or compatible with the character set for URLs embedded in client requests. The character set for the file system must provide for conversion of all possible characters in client requests. You must also specify the parameter `DAVOraNLS On`.

For example, assume that you are using Web folders on a system where the files have ShiftJIS characters and that the file system under dav_public is represented by the operating system in the JAPANESE_JAPAN.JA16SJIS character set, as shown in Figure 7–2.

Figure 7–2 OraDAV Access to File System with ShiftJIS Characters


WebDAV Client
http 1.1
HTTP Server
(Apache with OraDAV)
NLS_LANG = JAPANESE_JAPAN.JA16SJIS
File System
File names and file data contain ShiftJIS characters

In this case, you must do the following:

1. Set the NLS_LANG value to JAPANESE_JAPAN.JA16SJIS.
2. Include the following in the httpd.conf file.

```
<Location /dav_public>
  DAV On
  DAVOraNLS On
</Location>
```

DAV Parameter Information

This section describes some DAV parameters that you can set in the `httpd.conf` file.

DAVLockDB

To create the DAV lock database, add a `DAVLockDB` directive at the top-level of the configuration file (that is, outside a `<Directory>` or `<Location>` directive). The `DAVLockDB` directive should specify the name of a file that mod_dav will create. The directory in which the file is to be created must exist and, and the Oracle HTTP Server process must have write permission to it.

Note: The directory should not be on an NFS-mounted partition. mod_dav uses `flock/fcntl` to manage access to the database. Some operating systems cannot use these operations on an NFS-mounted partition.

In the following example:

```
DAVLockDB ORACLE_HOME/Apache/var/DAVLock
```

The DAV lock database will be stored in the *`ORACLE_HOME`*`/Apache/var` directory (which must be writable by the Oracle HTTP Server process). The file name will be `DAVLock` when mod_dav needs to create it. (Actually, mod_dav will create one or more files using this file name plus an extension).

The `DAVLockDB` directive can appear outside of any container or within a `<VirtualHost>` specification. It only needs to appear once, and a file extension should not be supplied.

DAVMinTimeout

The `DAVMinTimeout` directive specifies the minimum lifetime of a lock in seconds. If a client requests a lock timeout less than `DAVMinTimeout` value, then the `DAVMinTimeout` value is used and returned instead. For example, Microsoft's Web Folders defaults to a lock timeout of 2 minutes (120 seconds); however, you might decide to specify 10 minutes (600 seconds) instead, to reduce network traffic and the chance that the client might lose a lock due to network latency.

The DAVMinTimeout directive is optional, and may be used on a per-server or per-directory or location basis. The DAVMinTimeout directive takes a single positive integer. Because this value represents a minimum allowed, setting it to zero (0) will disable this feature. The default value for DAVMinTimeout is zero.

DAVDepthInfinity

A PROPFIND request with a Depth: Infinity header can impose a large burden on the server. These kinds of requests could “walk” the entire repository, returning information about each resource found. mod_dav builds the response in memory, so these kinds of requests can consume a lot of memory. (The memory is released at the end of the request, but the peak memory usage can be high.)

To prevent these kinds of requests, the DAVDepthInfinity directive is provided. It is a simple on / off directive, which can be used on a per-server, per-directory or location basis. The default value for this directive is off, meaning that these kinds of requests are not allowed. A value of on (that is, allowing depth infinity requests) makes it easier for denial of service attacks to occur. However, some clients, such as sitecopy, require a DAVDepthInfinity value of on.

Note: The WebDAV Working Group has stated that it is acceptable for DAV servers to refuse these kinds of requests. Properly written client software should not issue such requests, and you should not worry about disabling them.

DAVOraNLS

The DAVOraNLS directive provides globalization support for access to the local file systems. This directive specifies whether or not the file names in the file system need to go through conversion using the NLS_LANG setting. A value of Off, the default, means that no conversion is needed. A value of On means that the character set for the file system provides for conversion of all possible characters in client requests. For more information, see "Globalization Support Considerations with OraDAV" on page 7-19.

DAVOraWebCacheReadOnly

The DAVOraWebCacheReadOnly directive specifies whether or not Web Cache should be used by WebDAV clients. A value of Off, the default, means that Web Cache functions normally. A value of On prevents WebDAV clients from using Web Cache and thus potentially retrieving stale documents for editing; however, it does allow read-only activity by browsers and it does allow other clients to use Web Cache. For more information, see "Using Web Cache for Browsing Activities" on page 7-18.

LimitXMLRequestBody

mod_dav parses XML request bodies into memory. One technique used in denial of service attacks is to send a large request body at a mod_dav server. The Oracle HTTP Server defines a directive named LimitRequestBody, which limits all methods' request bodies. Unfortunately, this is not an effective mechanism for a mod_dav server because large PUT operations should be allowed.

To limit just the methods that have an XML request body, mod_dav provides the LimitXMLRequestBody directive. The default for this value is a compile-time constant, which is set to one million (1000000) bytes in the standard distribution. Setting the value to zero (0) will disable the size limit.

LimitXMLRequestBody may be set on a per-server or a per-directory or location basis, and takes a single non-negative integer argument.

Limit

The DAV and DAVLockDB directives are the only two configuration changes necessary to operate a DAV server. However, it is usually best to secure the site to be writable only by specific authorized users. This requires the use of the <Limit> directive. For example:

```
<Location /mypages>
DAV On
<Limit PUT POST DELETE PROPFIND PROPPATCH MKCOL COPY MOVE LOCK UNLOCK>
Require user greg
</Limit>
</Location>
```

This configuration will allow only authorized users to manipulate the site. However, it does allow them a bit more freedom than you may like. In particular, they may be able to place an .htaccess file into the target directory, altering your server configuration. The server may have already been configured to not read .htaccess files, but it is best to make sure. Also, you may want to disallow other options within the DAV-enabled directory -- CGI, symbolic links, server-side includes, and so on. Here is a modified configuration with the additional restrictions placed on it through the addition of `AllowOverride None` and `Options None`:

<Location /mypages>

```
DAV On
AllowOverride None
Options None
<Limit PUT POST DELETE PROPFIND PROPPATCH MKCOL COPY MOVE LOCK UNLOCK>
Require user greg
</Limit>
</Location>
<Location /mypages>
DAV On
AllowOverride None
Options None
<Limit PUT POST DELETE PROPFIND PROPPATCH MKCOL COPY MOVE LOCK UNLOCK>
Require user greg
</Limit>
</Location>
```

LimitExcept

Rather than using the `<Limit>` directive and specifying an exhaustive list of HTTP methods to secure, it is also possible to use the `<LimitExcept>` directive. This directive applies the access restrictions to all methods except for the methods listed. For example:

```
<Location /mypages>
 DAV On
 AllowOverride None
 Options None
 <LimitExcept GET HEAD OPTIONS>
 require user webadmin
 </LimitExcept>
</Location>
```

Choosing to use one or the other is a matter of preference. The <Limit> directive is precise and explicit, but the <LimitExcept> directive will automatically restrict methods that are added in the future.

PROPFIND Security

In the example configurations in the preceding sections on the `<Limit>` and `<LimitExcept>` directives, the PROPFIND method was limited, even though it is read-only. This is because the PROPFIND method can be used to list all the files in the DAV-enabled directory. For security reasons, it is probably best to protect the list of files from general read access.

An alternative would be to limit the PROPFIND to a group of people, a set of domains, or a set of hosts, while the methods that modify content are limited to just a few authors. This scenario allows, say, your company's employees to browse the files on the server, yet only a few people can change them. Anonymous (non-authenticated) visitors cannot browse or modify.

Finally, you can simply omit PROPFIND from the limits if your Web server is intended as a general, read-only repository of files. This allows anybody to arbitrarily browse the directories and then to fetch the files.

8

Frequently Asked Questions

This chapter provides answers to frequently asked questions on how to configure the Oracle HTTP Server to perform specialized useful functions.

See Also: "Frequently Asked Questions" in the Apache Server documentation.

Note: Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click on the hyperlinks.

1. How do I create an application-specific error page?

   Oracle HTTP Server has a default content handler for dealing with errors. You can use the `ErrorDocument` directive to override the defaults.

   > **See Also:** "ErrorDocument directive" in the Apache Server documentation.

2. How can I offer HTTPS to my ISP (virtual host) customers?

   For HTTP, Oracle HTTP Server supports two types of virtual hosts: name-based and IP-based. HTTPS supports only IP-based virtual hosts.

   If you are using IP-based virtual hosts for HTTP, then the customer has a virtual server listening on port 80 of a per-customer IP address. To provide HTTPS for these customers, simply add an additional virtual host per user listening on port 443 of that same per-customer IP address and use SSL directives, such as `SSLRequireSSL` to specify the per-customer SSL characteristics. Note that each customer can have their own wallet and server certificate.

   If you are using name-based virtual hosts for HTTP, each customer has a virtual server listening on port 80 of a shared IP address. To provide HTTPS for those customers, you can add a single shared IP virtual host listening on port 443 of the shared IP address. All customers will share the SSL configuration, including the wallet and ISP's server certificate.

3. How can I use the Oracle HTTP Server as a Web cache?

   You can use the Oracle HTTP Server as a Web cache by setting the `ProxyRequests` "on" and `CacheRoot` directives.

   > **See Also:** "ProxyRequests and CacheRoot directives" in the Apache Server documentation.

4. How can I configure the Oracle HTTP Server to use different language and character set versions of a document?

   You can use *multiviews*, a general name given to the Apache server's ability to provide language and character-specific document variants in response to a request.

   > **See Also:** "Multiviews" in the Apache Server documentation.

5. How do I configure the Oracle HTTP Server if I use Oracle9*i*AS Web Cache in front of it?

   You can use directives such as `ExpiresActive`, `ExpiresByType`, `ExpiresDefault`, to set the length of time that any cache existing between the client and the Web server will cache the returned Web pages.

   > **See Also:** "ExpiresActive, ExpiresByType, ExpiresDefault directives" in the Apache Server documentation.

6. When the Oracle HTTP Server is in front of a firewall, how should it send proxy sensitive requests to an Oracle HTTP Server behind a firewall?

   You should use the Proxy directives, and not the Cache directives, to send proxy sensitive requests across firewalls.

7. How can I use `<Directory>`, `<Location>`, `Alias`, and other directives to create a simple, distributed application name space that works across firewalls, clusters of application servers, and Web caches?

   The general idea is that all servers in a distributed Web site should agree on a single URL namespace. Every server will serve some part of that namespace, and will be able to redirect or proxy requests for URLs that it does not serve to a server that is "closer" to that URL. For example, your namespaces could be the following:

   ```
   /app1/login.html
   /app1/catalog.html
   app1/dologin.jsp
   /app2/orderForm.html
   /apps/placeOrder.jsp
   ```

   We could initially map this namespace to two Web servers by putting app1 on server1 and app2 on server2. Server1's configuration might look like the following:

```
Redirect permanent /app2 http://server2/app2
Alias /app1 /myApps/application1
<Directory /myApps/application1>
...
</Directory>
```

Server2's configuration is complementary. Now, if we decide to partition the namespace by content type (HTML on server, JSP on server2), we change server configuration and move files around, but we do not have to make changes to the application itself. The resulting configuration of server1 might look like the following:

```
RedirectMatch permanent (.*) \.jsp$ http://server2/$1.jsp
AliasMatch ^/app(.*) \.html$ /myPages/application$1.html
<DirectoryMatch "^/myPages/application\d">
...
</DirectoryMatch>
```

Note that the amount of actual redirection can be minimized by configuring a hardware load balancer like F5 system's BigIP to send requests to server1 or server2 based on the URL.

8. How do I protect my Web site from hackers?

There are many attacks, and new attacks are invented everyday. Following are some general guidelines for securing your site. You can never be really completely secure, but you can avoid being an easy target.

- Use a commercial firewall, such as Checkpoint FW-1 or Cisco PIX between your ISP and your Web server. Recognize, however, that not all hackers are outside your organization.
- Use switched ethernet to limit the amount of traffic a compromised server can sniff. Use additional firewalls between Web server machines and highly sensitive internal servers running database and enterprise applications.
- Remove unnecessary network services such as RPC, Finger, telnet from your server machine.
- Carefully validate all input from Web forms. Be especially wary of long input strings and input that contains non-printable characters, HTML tags, or javascript tags.
- Encrypt or randomize the contents of cookies that contain sensitive information. For example, it should be difficult to guess a valid sessionId to prevent a hacker from hijacking a valid session.

- Check often for security patches for all your system and application software, and install them as soon as possible. Be sure these patches come from bona fide sources; download from trusted sites and verify the cryptographic checksum.
- Use an intrusion detection package to monitor for defaced Web pages, viruses, and presence of “rootkits” that indicate hackers have broken in. If possible, mount system executables and Web content on read only file systems.
- Have a “forensic analysis” package on hand to capture evidence of a break in as soon as detected. This will aid in prosecution of the hackers.

A

Using the Oracle9*i*AS Proxy Plug-in

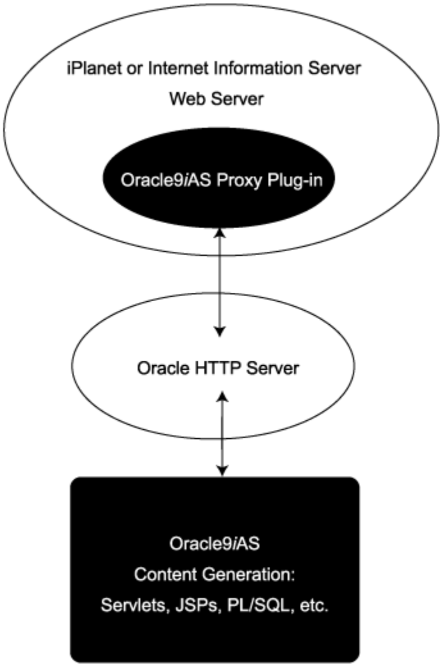
This appendix explains how the Oracle9*i*AS Proxy Plug-in enables you to use Oracle9*i*AS components in conjunction with a third-party HTTP listener. The Oracle9*i*AS Proxy Plug-in works with the Netscape *i*Planet Web Server Enterprise Edition (version 4.1 and 6.0) on UNIX and Windows NT systems, or the Microsoft Internet Information Server (IIS) (version 4.0 or 5.0) on Windows systems, to send requests to Oracle9*i*AS.

This appendix contains the following topics:

Overview

The Oracle 9iAS Proxy Plug-in is a reverse HTTP proxy. It forwards incoming HTTP requests to an Oracle9*i*AS instance as shown in Figure A–1.

Figure A–1 Oracle9iAS Proxy Plug-in


iPlanet or Internet Information Server
Web Server
Oracle9*i*AS Proxy Plug-in
Oracle HTTP Server
Oracle9*i*AS
Content Generation:
Servlets, JSPs, PL/SQL, etc.

This proxy logic is provided as a plug-in, a shared library that is loaded by the third-party HTTP listeners. The plug-in uses APIs provided with the third-party listeners to directly handle HTTP requests, in much the same way that modules (mods) are plugged into the Oracle HTTP Server.

The Oracle HTTP Server can mimic the address and port that the third-party listener is using. That is, when sending a request to the Oracle HTTP Server, the proxy can be configured to send a different Host: HTTP header than the actual hostname and port that the request is being sent to, so that downstream applications are shielded from the introduction of the reverse proxy.

Downloading the Plug-in

The plug-in is distributed on the Integration CD, and available on OTN at

http://otn.oracle.com

Installing the Plug-in

There is no installation procedure for the proxy plug-in. After downloading the plug-in, you need only place the configuration files and shared library in directories that the third-party listener can access.

Installing the Plug-in on UNIX Systems

The plug-in consists of a single shared library, `oracle_proxy.so`. To install the plug-in into the listener, simply place the library in a directory to which the listener has read and execute privileges.

Installing the Plug-in on Windows Systems

The plug-in consists of a single dll, `oracle_proxy.dll` for IIS, or `oracle_proxy_nes.dll` for Netscape. To install this plug-in, copy the .dll to a directory the listener can access.

Configuring the Plug-in

There are three configuration files for the Oracle9*i*AS Proxy Plug-in, one to control the proxy functionality and two to control the single sign-on functionality. The presence of the configuration file in the web server's file system makes the functionality active.

You may also need to modify configuration files specific to the third-party listener to install the plugin on to these listeners. Configuration instructions for those files are not included here.

Proxy Configuration File

This file must reside in a directory that is readable by the third-party listener. Described in detail in Table A–1, this file contains:

- Name value pairs that describe the servers that will be used to proxy requests to Oracle9*i*AS.
- Options for communicating with the servers.
- A set of rules that map URLs to the servers.

Table A–1 Proxy Configuration File Parameters

Parameter Name	Description
oproxy.serverlist	This parameter lists all of the server names that the plug-in will recognize. Parameter Type: string list Allowable Values: Comma separated list of server names, one for each Oracle HTTP Server to which requests will be sent. All servers in the serverlist must also be defined in the file. Default Value: None. At least one server name must be provided for the proxy to be functional. Example: `oproxy.serverlist=ias1,ias2`
oproxy.*servername*.hostname	This parameter defines the hostname to use when communicating with a specific server. Parameter Type: string Allowable Values: Valid hostname. Default Value: None. Example: `oproxy.ias1.hostname=www1.us.oracle.com`
oproxy.*servername*.port	This parameter defines the port to use when communicating with a specific server. Allowable Values: Valid port value. Default Value: 80 Example: `oproxy.ias1.port=7777`

Table A–1 Proxy Configuration File Parameters (Cont.)

Parameter Name	Description
oproxy.*servername*.alias	This parameter supports the mimicing feature of the proxy by defining the hostname and port that clients use to access the third-party HTTP listener. If defined, this value will be passed as the Host: HTTP header. If not defined, the hostname and port of the machine actually being communicated with will be sent. Parameter Type: string Allowable Values: host:port Default Value: oproxy.*servername.hostname*:oproxy.*servername.port* Example: `oproxy.ias1.alias=www.oracle.com:80`
oproxy.*servername*.urlrule	This parameter describes a URL or set of URLs that will be redirected to this server. A given server can have any number of urlrule properties assigned to it. Parameter Type: string Example: `oproxy.ias1.urlrule=/foo/*` Three types of rules can be used: exact match, context match, or suffix match. Exact matches: one URL is mapped to a server. For example: `oproxy.ias1.urlrule=/foo/bar/foo.html` would map only the URL /foo/bar/foo.html to be proxied to the server with the name ias1 (the details for the server ias1 are configured in the server config file).

Table A–1 Proxy Configuration File Parameters (Cont.)

Parameter Name	Description
	Context matches: a set of URLs with a common prefix or context are all mapped to a server. For example:
	oproxy.ias1.urlrule=/foo/*
	would map URLs beginning with /foo to the server with the name ias1.
	For context matches, you can use the stripcontext option with the urlrule parameter to send only the portion of the url *following* the wildcard to the server. The default for the stripcontext option is false, so you do not need to include it unless you are setting it to true. It is shown below for completeness of the example.
	Example:
	The following configuration:
	`oproxy.ias1.urlrule=/ias1/*`
	`oproxy.ias1.stripcontext=`**`false`**
	and the URL request:
	http://*hostname*/**ias1/header1.gif**
	retrieves
	ORACLE_HOME/Apache/Apache/htdocs/**ias1/header1.gif**
	The following configuration:
	`oproxy.ias1.urlrule=/ias1/*`
	`oproxy.ias1.stripcontext=`**`true`**
	and the URL request:
	http://*hostname*/**ias1/header1.gif**
	retrieves:
	ORACLE_HOME/Apache/Apache/htdocs/**header1.gif**
	Suffix matches: all files with a common file extension are mapped to a server. For example:
	oproxy.ias1.urlrule=/*.jsp
	would map all of the URLs that end in .jsp to the server ias1. This can be combined with the context rule to have something like:
	/foo/bar/*.jsp
	so that only URLs that start with /foo/bar and end in .jsp would be proxied.

Table A–1 Proxy Configuration File Parameters (Cont.)

Parameter Name	Description
	When multiple rules apply to the same URL, the following precedence applies: 1. Exact matches 2. Longest context match plus suffix match 3. Longest context match Some examples of the precedence are: /foo/bar/index.html would take precedence over /foo/bar/* /foo/bar/*.jsp would take precedence over /foo/bar/* /foo/bar/* would take precedence over /foo/*

Defining Proxy Plug-in Behavior

In the proxy configuration file, you define which servers and URLs to proxy to the plugin.

1. In the first line of the file, specify the list of all the servers that can be used by the plugins. For example:

```
oproxy.serverlist=ias1,ias2
```

2. Set the relevant properties (hostname, port, and server alias) for each server. For example:

```
oproxy.ias1.hostname=myhost.us.oracle.com
oproxy.ias1.port=7777
oproxy.ias1.alias=www.oracle.com
```

   The hostname must be provided. If you do not specify the port, 80 is assigned. If an alias value is not given, the combination of the hostname and port given are used. The alias enables the back end server to receive requests that have an HTTP Host: header that looks exactly like the one the client delivers to the third-party listener.

3. Set the urlrule parameter to specify redirection between servers. For example, the rule:

   ```
   oproxy.ias1.urlrule=/*
   ```

   maps all incoming requests to be proxied to the web server on the server ias1. These rules can be of three forms, exact URL, context match, or extension-based. An exact match maps exactly one URL to a server, for example:

   ```
   oproxy.ias1.urlrule=/my/path/index.html
   ```

   maps only accesses to /my/path/index.html for proxying. An example of a context rule is:

   ```
   oproxy.ias1.urlrule=/app1/*
   ```

   which maps any URL beginning with /app1.

   An extension-based rule, such as:

   ```
   oproxy.ias1.urlrule=/*.jsp
   ```

   maps any URL ending with .jsp.

   All requests sent to a mapped URL are proxied via HTTP/1.1 to the specified server.

Configuring the *i*Planet Listener to Use the Proxy Plug-in

This section provides proxy plug-in configuration instructions for the Netscape *i*Planet Enterprise Server listener on UNIX and Windows NT systems.

> **Notes:** If you are configuring the *i*Planet listener on Windows NT, use forward slashes (/) in all paths.
>
> The default configuration files for Netscape *i*Planet route all incoming requests for the URI `/servlet` to the Netscape servlet handler. The Oracle Proxy Plug-in does not override settings Netscape configuration settings. You must ensure that the URL mappings to the Oracle9*i*AS Proxy Plug-in are distinct from the URL mappings to the Netscape servlet engine.

1. Open the `magnus.conf` file (version 6) or `obj.conf` (version 4) in the Netscape listener `/config` directory.
2. Add the load-modules line:

   For UNIX:

   ```
   Init fn="load-modules" shlib="/path/oracle_proxy.so" funcs=op_init,op_
   objecttype,op_service
   ```

   For Windows NT:

   ```
   Init fn="load-modules" shlib="/path/oracle_proxy_nes.dll" funcs=op_init,op_
   objecttype,op_service
   ```

   where `/path/` is the path to the shared library for the plug-in. This line tells the listener where the proxy shared library is, and which functions are exposed by this library.
3. Add the configuration parameters line:

   ```
   Init fn="op_init" server_defs="/path/config/servers" log_
   file="/path/logs/oproxy.log" log_level=error
   ```

   where `/path/` is the path to the server definition and log files. The server definition file contains all of the configuration information for the servers that the proxy plug-in can communicate with. A log file and log level to log messages from the plug-in can also be specified (optional).
4. Add the following line to the <Object name=default> section of the `obj.conf` file, before all other lines beginning with the word ObjectType:

   ```
   ObjectType fn=op_objecttype
   ```

5. Add the following line before all other lines that begin with the word Service:

   ```
   Service type="oracle/proxy" fn="op_service"
   ```

6. Start the listener using the GUI or the shell script.

Configuring the IIS Listener to Use the Proxy Plug-in

This section provides proxy plug-in configuration instructions for the IIS listener on Windows systems. To configure the plug-in:

1. Create a new registry key named `HKEY_LOCAL_MACHINE\SOFTWARE\Oracle\IIS Proxy Adapter`
2. Specify the location of your server definition file by adding a string value with the name server_defs, and a value pointing to the location of your definition file.
3. (Optional) Specify `log_file` and `log_level`:
   a. Add a string value with the name `log_file` and the desired location of the log file (for example, `d:\proxy\proxy.log`)
   b. Add a string value with the name `log_level` and a value for the desired log level. Valid values are debug, inform, error and emerg.
4. Using the IIS management console, add a new virtual directory to your IIS web site with the same physical path as that of `oracle_proxy.dll`. Name the directory `oproxy` and give it execute access.
5. Using the IIS management console, add `oracle_proxy.dll` as a filter in your IIS web site. The name of the filter should be `oproxy` and its executable must point to the directory containing `oracle_proxy.dll` (for example, `d:\proxy\oracle_proxy.dll`).
6. Restart IIS (stop and then start the IIS Server), ensuring that the oproxy filter is marked with a green upward arrow.

> **Note:** To restart IIS, you must stop all of the IIS services through the control panel, or restart the computer. This is the only way to ensure that the DLL is reloaded (restarting IIS through the management console is not sufficient).

Using Single-Sign On with the Plug-in

In addition to the proxy functionality, the proxy plug-in includes the same functionality provided by mod_osso to the Oracle HTTP Server to support single sign-on. With single sign-on, users need authenticate only once to the web server; thereafter, the user name and password are relayed invisibly to Oracle9*i*AS applications.

If you want to use single sign-on functionality, you must first install the database Required Support Files libraries (v9.0.1.3) in order to get the necessary security libraries.

Configuring Single Sign-On Plug-Ins

Oracle Single Sign-On (OSSO) can be used with *i*Planet and IIS web servers. OSSO implements the basic functionalities of mod_osso, the single sign-on module for the Oracle HTTP Server. Features not implemented in the current release include:

- Dynamic authentication
- Global logout
- Basic authentication for legacy applications.

Both *i*Planet and IIS OSSO plug-ins use the same formatted `osso_plugin.conf` file. You must create this file and put it in a location that the listener can access.

OSSO Configuration File Examples

This section contains examples of entries from the OSSO configuration file `osso_plugin.conf`.

Note: The osso_plugin.conf file must be referenced by the `obj.conf` file, configuration parameters line, as follows:

```
Init fn="osso_init" osso_properties="/path/config/osso_
plugin.conf" log_file="/path/debug.log" log_level=error
```

This line can also specify a log file and log level to log messages from the plug-in (optional).

```
LoginServerFile = "/path/config/sso_conf"
HardTimeOut = 1000
```

In the LoginServerFile directive above, *path* is the *i*Planet server root directory.

```
<OSSO /private/hello.html>
 IpCheck = true;
 HardTimeOut = 10000
</OSSO>

<OSSO /private1/*>
 IpCheck = false
</OSSO>

<OSSO /private2/*.jsp>
 IpCheck = false
 HardTimeOut = 100
</OSSO>
```

OSSO directives

This section describes the OSSO directives.

LoginServerFile Specifies the location of the Single Sign-On Server configuration file that provides all the information about the Single Sign-On Server, such as version, cipher key, etc. This Single Sign-On Server configuration file is obfuscated. See "Obtaining an Obfuscated Single Sign-On Server Configuration File" on page A-16.

Currently, only one Single Sign-On Server is allowed for all the protected resources, so you cannot use this directive on a per-resource basis. You must provide one and only one Single Sign-On Server configuration file.

IpCheck Specifies whether OSSO should check the IP address of each request when it examines the cookie. Valid values are true and false. Setting IpCheck to true prevents cookies being stolen.

Example:

```
IpCheck = true
```

Note: Set IpCheck to false if you have a proxy server or firewall between your *i*Planet server and your clients' browser.

HardTimeout Specifies the lifetime (in seconds) of the cookie for the protected resource. After this amount of time, the user will have to authenticate again.

Example:

```
HardTimeout = 100
```

Specifying Protected URIs

You can specify URIs to protect in several ways:

Exact Context This option identifies a particular file as a protected resource, for example:

```
/examples/Hello.html
```

Context Based This option identifies a directory as a protected resource, for example:

```
/examples/*
```

Context and Suffix This option identifies files with a certain extension in a particular directory as a protected resource, for example:

```
/examples/*.jsp
```

Configuring the *i*Planet Listener for Single Sign-on

This section provides instructions on configuring the *i*Planet Enterprise Server listener to use the OSSO plug-in.

1. Open the `obj.conf` file in the Netscape listener /config directory.
2. Add the load-modules line:

   ```
   Init fn="load-modules" shlib="/path/oracle_proxy.so" funcs="osso_
   init,oracle_single_sign_on,osso_redirect,osso_success_service"
   ```

   where /path/ is the path to the shared library for the plug-in. This line tells the listener where the proxy shared library is, and which functions are exposed by this library.

3. Add the configuration parameters line:

   ```
   Init fn="osso_init" osso_properties="/path/config/osso_plugin.conf" log_
   file="/path/debug.log" log_level=error
   ```

   where /path/ is the *i*Planet server root directory. This line contains the location of an OSSO definition file that contains all of the configuration information for the servers that the OSSO plug-in will use. This line can also specify a log file and log level to log messages from the plug-in (optional).

4. Add the following line to the <Object name=default> section of the `obj.conf` file, before all other lines:

   ```
   AuthTrans fn="oracle_single_sign_on"
   ```

5. Add the following line to the Object name=default section before all other lines that begin with the word Service:

   ```
   Service type="oracle/sso_redirect" fn="osso_redirect_service"
   ```

6. Add the following lines:

   ```
   <Object ppath="/path/osso_login_success">
     Service fn="osso_success_service"
   </Object>
   ```

   where path is the path of your document root (e.g., home/me/iplanet/docs/). This should be the same path as named in

   ```
   NameTrans fn=document-root root=/.../iplanet/docs
   ```

7. Change the LD_LIBRARY_PATH variable in your start script to include the location of libclntsh.so.

8. Start the listener using the GUI or the shell script.

Usage Notes for Netscape/*i*Planet Enterprise Server Version 6.0

For version 6.0, the same shared library can be used as with version 4.1. The configuration is virtually the same, but the configuration files for Netscape have changed slightly in version 6.0. In this version, the two lines beginning with `Init` that need to be added must be added at the end of the `magnus.conf` file rather than to the `obj.conf` file. The other two lines that should be added to obj.conf remain the same.

Configuring the IIS Listener for Single Sign-On

This section provides instructions on configuring the IIS Listener to use the OSSO plug-in. The plugin consists of a single dll, `oracle_osso.dll`. To install the plugin, copy the .dll to the computer on which IIS resides and perform the following steps:

1. Edit your registry to create a new registry key named `HKEY_LOCAL_MACHINE\SOFTWARE\Oracle\IIS OSSO Adapter`.
2. Specify the location of your osso configuration file by adding a string value with the name `cfg_file` and a value pointing to the location of your definition file.
3. (Optional) Specify a log_file and log_level:
   a. Add a string value with the name log_file and the desired location of the log file (for example, `d:\osso\osso.log`)
   b. Add a string value with the name log_level and a value for the desired log level. Valid values are debug, inform, error and emerg.
4. Using the IIS management console, add a new virtual directory to your IIS web site with the same physical path as that of `oracle_osso.dll`. Name the directory `osso` and give it execute access.
5. Using the IIS management console, add `oracle_osso.dll` as a filter in your IIS web site. The name of the filter should be `osso` and its executable must point to the directory containing `oracle_osso.dll` (for example, `d:\osso\oracle_osso.dll`).
6. Restart IIS (stop and then start the IIS Server), ensuring that the oproxy filter is marked with a green up-pointing arrow.

Note: To restart IIS, you must stop all of the IIS services through the control panel, or restart the computer. This is the only way to ensure that the DLL is reloaded (restarting IIS through the management console is not sufficient).

Obtaining an Obfuscated Single Sign-On Server Configuration File

Follow these steps to obtain an obfuscated Single Sign-On Server configuration file for the *i*Planet server.

1. Install the Oracle9*i*AS core components on the computer on which the *i*Planet server resides.

2. Register the *i*Planet server with a Single Sign-On server using the command below. If entered line by line on a terminal, each line must end with the backslash character. Command arguments are shown in italics; you must supply valid values. Arguments are described in Table A–2, "SSO Registrar Command Arguments" on page A-17.

```
$ORACLE_HOME/jdk/bin/java -jar $ORACLE_HOME/sso/lib/ossoreg.jar \
-oracle_home_path $ORACLE_HOME \
-host login server database hostname \
-port login server database port \
-sid login server database sid \
-site_name name of iplanet server \
-success_url http://iplanet server hostname:port/osso login success url \
-cancel_url http://iplanet server hostname:port/cancel url \
-logout_url http://iplanet server hostname:port/osso logout success url \
-home_url http://iplanet server hostname:port/home url \
[-admin_id administrator user id] \
[-admin_info administrator information] \
-config_mod_osso TRUE \
-virtualhost \
-u userid \
-sso_server_version v1.2 \
-text_file $ORACLE_HOME/cleartext sso configuration file \
-config_file $ORACLE_HOME/Apache/Apache/conf/osso/sso_conf \
```

3. Copy the generated `sso_conf` file to the *i*Planet root directory

4. Ensure that the LoginServerFile directive in the `osso_plugin.conf` file specifies the generated `sso_conf` file.

SSO Registrar Command Arguments

Table A–2 lists the command arguments for the SSO Registrar.

Table A–2 SSO Registrar Command Arguments

Argument	Description
-oracle_home_path	Absolute path to the Oracle home.
-site_name	Name of the site, typically expressed as the contiguous string *host:port*.
-success_url	URL to the routine responsible for establishing the partner application session and session cookies. The URL is: http://*iplanet server hostname*:*port*/osso_login_success
-cancel_url	URL to which the SSO server redirects when the user cancels authentication.
-admin_id	User name of the mod_osso administrator. This argument is optional.
-admin_info	Information associated with the administrator's user name, such as e-mail address. This argument is optional.
-config_ mod_osso	If set to TRUE, this parameter indicates that the application being registered is mod_osso. This argument is necessary to generate the `sso_conf` file.

Table A–2 SSO Registrar Command Arguments

Argument	Description
-u	Specifies the user name under which the Oracle HTTP Server was started. In the case of the *i*Planet server, it is the name of the user under which the *i*Planet server will run (specified during *i*Planet server configuration). Note the difference between the -u value specified for the Oracle HTTP Server and that specified for the *i*Planet server: ■ For the Oracle HTTP Server, -u is the user that started the server. ■ For the *i*Planet server, -u is the user under which the server answers client requests. You can specify `-u root` for the Oracle HTTP Server. However, Oracle Corporation recommends against specifying `-u root when` configuring an iPlanet server (and, by implication, not to use `-u root` when registering the iPlanet server with the SSO server. This argument is optional. The default is the user name under which the SSORegistrar tool is run. In this case, the Oracle HTTP Server must be started under that user name, and the iPlanet server must be configured to answer client requests under that name.
-sso_server_version	Must be set to `v1.2` in Oracle9*i*AS Release 2.
-virtualhost	Must be specified when registering an *i*Planet server. However, it is optional when registering Oracle HTTP Server. If it is left unspecified, the Oracle HTTP server belongs to a cluster.
-text_file	Specifies a path to a temporary version of the mod_osso configuration file. It is optional, and the default location is: `ORACLE_HOME/Apache/Apache/conf/osso/osso.txt.`
-config_file	Specifies a path to the mod_osso configuration file. It must be specified if -virtualhost is present. For this release, its value can only be set to: `ORACLE_HOME/Apache/Apache/conf/osso/sso_osso.`

Proxy Plug-In Usage Notes

This section highlights development and usage practices to consider when developing an application that will run behind the Oracle9*i*AS Proxy Plug-in. Some of these also have relevance when enabling an application to run behind Oracle9*i*AS Web Cache.

- **Check for configurations based on the Oracle HTTP Server being the entry point into the network.**

  This is usually only relevant if an application has a module that plugs directly into the Oracle HTTP Server. Specifically, look for dependencies on obtaining information about the client based on the connection made to the Oracle HTTP Server (such as using the SSL certificate for authentication). Currently, SSL is not supported, so even if the client uses SSL to connect to the third-party listener, an unencrypted HTTP message will be sent from the third-party listener to the Oracle HTTP Server. This means that client certificates will not be available to components that reside behind the plug-in. The environment variable REMOTE_ADDR has been specifically preserved when Oracle9*i*AS Proxy Plug-in and Web Cache are used, but other client information may, in practice, represent the machine on which the proxy resides rather than the actual client host. These behaviors must be discovered and eliminated in cases where the Oracle HTTP Server is not the external listener for Oracle9*i*AS.

- **Avoid returning non-relative links in HTML, that is, avoid embedding host names into HTML unless the link is external to the website.**

  This includes static HTML pages, dynamic pages generated by servlets, JSPs, PL/SQL, etc. Examine all code that obtains the server name of the Oracle HTTP Server to ensure that it is not embedding the server name into pages that are sent back to the client. To test for this behavior, use a “spider” application that traverses all links in a web site. Open source tools with this functionality are available.

- **Avoid returning host and port information in applications (such as applets or javascript) downloaded to the client.**

  If you have an application that uses browser-based code, ensure that the code does not contain the hostname and port of the Oracle HTTP Server that actually delivers the content. Instead, it must have the actual client-accessible address used by the third-party listener.

- **Ensure that all URLs within an application can be easily mapped to a set of rules that the proxy can use.**

In order to successfully proxy all requests for an application, the Oracle9*i*AS Proxy Plug-in must have a complete description of the URL space for that application. Each Oracle9*i*AS application must describe the set of rules necessary to configure the plug-in for that application. This set of rules must include all URLs that the application could generate. If an application generates a URL that is not described by the proxy urlrule parameters, the request will be served by the third-party HTTP listener, and a “document not found” error may occur (or, worse, a document other then the intended document may be delivered to the client).

Developers of applications that use common top level directories (such as a reliance on mapping /images) should be prepared to:

- Change these common links to something that won’t conflict with applications that might already be deployed on the third-party listener, or
- Instruct the user to copy the necessary content to the third-party listener’s directory structure. For performance reasons, it is a good idea to have the third-party listener handle static .gif and .jpg files anyway, but it requires extra effort.

Troubleshooting

This section describes common problems and possible solutions.

Listener Fails To Start

- Ensure that you have the newest version of the Oracle9*i*AS Proxy Plug-in.
- Verify that your listener configuration is set up correctly. (The IIS listener may need to be restarted in order to make the filter work properly.) A server definition file must exist and cannot be empty.
- Check for problems in the server definition file. Each server in the *serverlist* line must be defined later in the file, and you must have at least one server defined. If a server name is listed but not defined, the Listener may not start (although the reverse is not true). Ensure that there are no typographical errors or missing quotes in the server definition file.
- **For *i*Planet 6.0 on UNIX and Windows:** Ensure that Init lines are added to the `magnus.conf` file and ObjectType and Service lines are added to the `obj.conf` file.

Listener Returns Incorrect URLs

- Verify that changes to the server definition file have been saved and the listener has been restarted.
- Ensure that there are no typographical errors in the server definition file.
- Ensure that the urlrule parameter is set up correctly, and consider whether the stripcontext option should be set to true.
- Verify that the *serverlist* line in the server definition file specifies the back-end server you're trying to reach.
- Verify that the back-end server is running, and that the file you're attempting to retrieve exists and is accessible on the back-end server.
- Verify that the host, port and urlrule parameters in the server definition file target the correct area on the back-end server.
- Ensure that client requests are being sent to the correct port on the third-party listener machine.
- Check the listener log files, the proxy log (may need to be turned on in “debug” mode, and may require restarting the listener), and the back-end server logs to verify that requests are getting through.

Changes Made to Server Definition File Are Not Reflected

- Ensure that you have saved the server definition file and restarted the listener.
- **For IIS:** To pick up the changes, you must stop and start the WWW Publishing Service from the Control Panel. This takes a few minutes.

IIS Listener Displays Incomplete Pages or Garbled Characters

- Do not display an IIS pages with a Netscape browser.

Parsing Error Occurs with *i*Planet 6.0

If you try to change the ports or turn on security (for SSL), the server may return the error message “Unable to parse magnus.conf”.

- Remove any comments and added lines preceding and following the Init lines in the `magnus.conf` file.

“File Not Found” Error Occurs

If you are using a context-based urlrule parameter to retrieve a file that is known to exist, and the listener returns “Not Found”, you probably need to set “stripcontext=true”. See the oproxy.*servername*.urlrule parameter, stripcontext option, in the table "Proxy Configuration File Parameters" on page A-4.

Partial URL Requests Return Unexpected Results

The IIS and *i*Planet servers auto-complete URLs differently. Requests of "http://serverName", “http://serviceman/”, and “http://serviceman/index.html” do not necessarily return the same results on different platforms. The oproxy.*servername*.urlrule parameter (described in the table "Proxy Configuration File Parameters" on page A-4) can be used to work around this problem.

*i*Planet Server Returns “Server Error” With “/servlet” Request

The default *i*Planet configuration maps any URL requests to “/servlet” to its own servlet handler. You must edit the server definition file, or change the *i*Planet configuration to correct this.

Server Returns Page With Broken Image Links

If you use an exact urlrule parameter (e.g. “urlrule=/*.html”) in the server definition file (or a similar scenario), the server will retrieve the specified page, but all other links are forbidden to the user (including inline images in the page). (If you use an exact urlrule with stripcontext=true, a "Server Error” is returned.)

Unexpected Pages Are Displayed

Clear the memory cache in your client browser. Earlier versions of Netscape and IE cache pages even when told to retrieve the page every time, when no memory is allocated for caching (you may need to restart the browser to get this behavior to work). If you see a page you're not expecting, try refreshing or reloading the page.

REMOTE_ADDR Contains Unexpected IP Address

The REMOTE_ADDR field usually contains the IP address of the client machine. In some URL request cases, if there is a proxy server in the environment, the field may contain the IP address of the proxy server.

Redirects Go To Network Entry Point

If the back-end server returns a redirect to the entry point of the network, do one of the following:

Solution 1 (Preferred)

Set the following directives in the `httpd.conf` file:

```
UseCanonicalName On
ServerName name of listener computer
Port port of listener computer
```

Solution 2

Set the following directives in the `httpd.conf` file:

```
UseCanonicalName port
Port port of Listener computer
```

Edit the proxy plug-in server configuration file:

```
oproxy.serverName.alias=name of listener computer:port of istener computer
```

SSL Requests Yield Unexpected Results

The proxy plug-in supports SSL connections made between the client and the proxy computer, but does not support SSL connections between the proxy and the back-end server. To implement the latter, set up the listener to receive SSL connections and start the back-end server in non-SSL mode. No changes to the proxy configuration are needed.

B

Third Party Licenses

This appendix includes the Third Party License for all the third party products included with Oracle9*i* Application Server. Topics include:

- Apache HTTP Server
- Apache JServ
- Perl
- mod_dav
- FastCGI

Apache HTTP Server

Under the terms of the Apache license, Oracle is required to provide the following notices. However, the Oracle program license that accompanied this product determines your right to use the Oracle program, including the Apache software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the Apache software is provided by Oracle “AS IS” and without warranty or support of any kind from Oracle or Apache.

The Apache Software License

```
/* ====================================================================
 * The Apache Software License, Version 1.1
 *
 * Copyright (c) 2000 The Apache Software Foundation.  All rights
 * reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in
 *    the documentation and/or other materials provided with the
 *    distribution.
 *
 * 3. The end-user documentation included with the redistribution,
 *    if any, must include the following acknowledgment:
 *       "This product includes software developed by the
 *        Apache Software Foundation (http://www.apache.org/)."
 *    Alternately, this acknowledgment may appear in the software itself,
 *    if and wherever such third-party acknowledgments normally appear.
 *
 * 4. The names "Apache" and "Apache Software Foundation" must
 *    not be used to endorse or promote products derived from this
 *    software without prior written permission. For written
 *    permission, please contact apache@apache.org.
 *
 * 5. Products derived from this software may not be called "Apache",
 *    nor may "Apache" appear in their name, without prior written
 *    permission of the Apache Software Foundation.
 *
 * THIS SOFTWARE IS PROVIDED ‘‘AS IS’’ AND ANY EXPRESSED OR IMPLIED
 * WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
 * OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
 * DISCLAIMED.  IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR
 * ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF
 * USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
 * ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
 * OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT
 * OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 * ====================================================================
 *
 * This software consists of voluntary contributions made by many
 * individuals on behalf of the Apache Software Foundation.  For more
 * information on the Apache Software Foundation, please see
 * <http://www.apache.org/>.
 *
 * Portions of this software are based upon public domain software
 * originally written at the National Center for Supercomputing Applications,
 * University of Illinois, Urbana-Champaign.
 */
```

Apache JServ

Under the terms of the Apache license, Oracle is required to provide the following notices. However, the Oracle program license that accompanied this product determines your right to use the Oracle program, including the Apache software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the Apache software is provided by Oracle “AS IS” and without warranty or support of any kind from Oracle or Apache.

Apache JServ Public License

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistribution of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistribution in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- All advertising materials mentioning features or use of this software must display the following acknowledgment:

  This product includes software developed by the Java Apache Project for use in the Apache JServ servlet engine project (http://java.apache.org/).
- The names “Apache JServ”, “Apache JServ Servlet Engine” and “Java Apache Project” must not be used to endorse or promote products derived from this software without prior written permission.
- Products derived from this software may not be called “Apache JServ” nor may “Apache” nor “Apache JServ” appear in their names without prior written permission of the Java Apache Project.
- Redistribution of any form whatsoever must retain the following acknowledgment:

  This product includes software developed by the Java Apache Project for use in the Apache JServ servlet engine project (http://java.apache.org/).

THIS SOFTWARE IS PROVIDED BY THE JAVA APACHE PROJECT “AS IS” AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE JAVA APACHE PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Perl

Perl Kit Readme

This program is free software; you can redistribute it and/or modify it under the terms of either:

a. the GNU General Public License as published by the Free Software Foundation; either version 1, or (at your option) any later version, or

b. the “Artistic License” which comes with this Kit.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See either the GNU General Public License or the Artistic License for more details.

You should have received a copy of the Artistic License with this Kit, in the file named “Artistic”. If not, I'll be glad to provide one.

You should also have received a copy of the GNU General Public License along with this program in the file named “Copying”. If not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307, USA or visit their web page on the internet at http://www.gnu.org/copyleft/gpl.html.

For those of you that choose to use the GNU General Public License, my interpretation of the GNU General Public License is that no Perl script falls under the terms of the GPL unless you explicitly put said script under the terms of the GPL yourself. Furthermore, any object code linked with perl does not automatically fall under the terms of the GPL, provided such object code only adds definitions of subroutines and variables, and does not otherwise impair the resulting interpreter from executing any standard Perl script. I consider linking in C subroutines in this manner to be the moral equivalent of defining subroutines in the Perl language itself. You may sell such an object file as proprietary provided that you provide or offer to provide the Perl source, as specified by the GNU General Public License. (This is merely an alternate way of specifying input to the program.) You may also sell a binary produced by the dumping of a running Perl script that belongs to you, provided that you provide or offer to provide the Perl source as specified by the GPL. (The fact that a Perl interpreter and your code are in the same binary file is, in this case, a form of mere aggregation.) This is my interpretation of the GPL. If you still have concerns or difficulties understanding my intent, feel free to contact me.

Of course, the Artistic License spells all this out for your protection, so you may prefer to use that.

Perl Artistic License

The “Artistic License”

Preamble

The intent of this document is to state the conditions under which a Package may be copied, such that the Copyright Holder maintains some semblance of artistic control over the development of the package, while giving the users of the package the right to use and distribute the Package in a more-or-less customary fashion, plus the right to make reasonable modifications.

Definitions:

“Package” refers to the collection of files distributed by the Copyright Holder, and derivatives of that collection of files created through textual modification.

“Standard Version” refers to such a Package if it has not been modified, or has been modified in accordance with the wishes of the Copyright Holder as specified below.

“Copyright Holder” is whoever is named in the copyright or copyrights for the package.

“You” is you, if you're thinking about copying or distributing this Package.

“Reasonable copying fee” is whatever you can justify on the basis of media cost, duplication charges, time of people involved, and so on. (You will not be required to justify it to the Copyright Holder, but only to the computing community at large as a market that must bear the fee.)

“Freely Available” means that no fee is charged for the item itself, though there may be fees involved in handling the item. It also means that recipients of the item may redistribute it under the same conditions they received it.

1. You may make and give away verbatim copies of the source form of the Standard Version of this Package without restriction, provided that you duplicate all of the original copyright notices and associated disclaimers.
2. You may apply bug fixes, portability fixes and other modifications derived from the Public Domain or from the Copyright Holder. A Package modified in such a way shall still be considered the Standard Version.

3. You may otherwise modify your copy of this Package in any way, provided that you insert a prominent notice in each changed file stating how and when you changed that file, and provided that you do at least ONE of the following:

    a. place your modifications in the Public Domain or otherwise make them Freely Available, such as by posting said modifications to Usenet or an equivalent medium, or placing the modifications on a major archive site such as uunet.uu.net, or by allowing the Copyright Holder to include your modifications in the Standard Version of the Package.

    b. use the modified Package only within your corporation or organization.

    c. rename any non-standard executables so the names do not conflict with standard executables, which must also be provided, and provide a separate manual page for each non-standard executable that clearly documents how it differs from the Standard Version.

    d. make other distribution arrangements with the Copyright Holder.

4. You may distribute the programs of this Package in object code or executable form, provided that you do at least ONE of the following:

    a. distribute a Standard Version of the executables and library files, together with instructions (in the manual page or equivalent) on where to get the Standard Version.

    b. accompany the distribution with the machine-readable source of the Package with your modifications.

    c. give non-standard executables non-standard names, and clearly document the differences in manual pages (or equivalent), together with instructions on where to get the Standard Version.

    d. make other distribution arrangements with the Copyright Holder.

5. You may charge a reasonable copying fee for any distribution of this Package. You may charge any fee you choose for support of this Package. You may not charge a fee for this Package itself. However, you may distribute this Package in aggregate with other (possibly commercial) programs as part of a larger (possibly commercial) software distribution provided that you do not advertise this Package as a product of your own. You may embed this Package's interpreter within an executable of yours (by linking); this shall be construed as a mere form of aggregation, provided that the complete Standard Version of the interpreter is so embedded.

6. The scripts and library files supplied as input to or produced as output from the programs of this Package do not automatically fall under the copyright of this Package, but belong to whoever generated them, and may be sold commercially, and may be aggregated with this Package. If such scripts or library files are aggregated with this Package via the so-called “undump” or “unexec” methods of producing a binary executable image, then distribution of such an image shall neither be construed as a distribution of this Package nor shall it fall under the restrictions of Paragraphs 3 and 4, provided that you do not represent such an executable image as a Standard Version of this Package.

7. C subroutines (or comparably compiled subroutines in other languages) supplied by you and linked into this Package in order to emulate subroutines and variables of the language defined by this Package shall not be considered part of this Package, but are the equivalent of input as in Paragraph 6, provided these subroutines do not change the language in any way that would cause it to fail the regression tests for the language.

8. Aggregation of this Package with a commercial distribution is always permitted provided that the use of this Package is embedded; that is, when no overt attempt is made to make this Package's interfaces visible to the end user of the commercial distribution. Such use shall not be construed as a distribution of this Package.

9. The name of the Copyright Holder may not be used to endorse or promote products derived from this software without specific prior written permission.

10. THIS PACKAGE IS PROVIDED “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The End

mod_dav

mod_dav has been licensed to Oracle free of charge by Greg Stein under a license similar to the Apache Software Foundation license. The following copyright notice applies to mod_dav and Oracle's use of mod_dav:

Copyright © 1998-2001 Greg Stein. All rights reserved.

The following notices also apply:

This product includes software developed by Greg Stein <gstein@lyra.org> for use in the mod_dav module for Apache (http://www.webdav.org/mod_dav/).

The Greg Stein software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or Greg Stein. Under the terms of the license between Oracle and Greg Stein, Oracle is required to provide these notices. Note, however, that the Oracle program license that accompanied this product determines your right to use the Oracle program, including the Greg Stein software, and the terms contained in these notices do not change those rights.

The following license terms also apply:

1. Development License. Oracle can use and modify the mod_dav code.
2. Distribution Right. Oracle can distribute the mod_dav code in source or object form, royalty-free, subject to the following requirements:
   - Oracle needs to include in the documentation for iAS a copy of the mod_dav license terms.
   - Oracle must include the following copyright notice in the documentation for iAS: “Copyright © 1998-2001 Greg Stein. All rights reserved.” and the following acknowledgment: “This product includes software developed by Greg Stein <gstein@lyra.org> for use in the mod_dav module for Apache (http://www.webdav.org/mod_dav/).”
   - Oracle must retain all copyright notices, license terms, disclaimers and acknowledgments in the source code.
   - Oracle may not call products derived from the mod_dav code “mod_dav”, and “mod_dav” may not appear in any Oracle product name without the prior written permission of Greg Stein.
3. No Intellectual Property Protection. Oracle gets no IP warranty and no right to indemnification for IP infringement claims.

4. Unlimited Liability. Oracle's liability is unlimited if Oracle fails to comply with these license terms.

FastCGI

FastCGI Developer's Kit License

This FastCGI application library source and object code (the "Software") and its documentation (the "Documentation") are copyrighted by Open Market, Inc ("Open Market"). The following terms apply to all files associated with the Software and Documentation unless explicitly disclaimed in individual files.

Open Market permits you to use, copy, modify, distribute, and license this Software and the Documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this Software and Documentation may be copyrighted by their authors and need not follow the licensing terms described here. If modifications to this Software and Documentation have new licensing terms, the new terms must be clearly indicated on the first page of each file where they apply.

OPEN MARKET MAKES NO EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE SOFTWARE OR THE DOCUMENTATION, INCLUDING WITHOUT LIMITATION ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL OPEN MARKET BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY DAMAGES ARISING FROM OR RELATING TO THIS SOFTWARE OR THE DOCUMENTATION, INCLUDING, WITHOUT LIMITATION, ANY INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES OR SIMILAR DAMAGES, INCLUDING LOST PROFITS OR LOST DATA, EVEN IF OPEN MARKET HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS". OPEN MARKET HAS NO LIABILITY IN CONTRACT, TORT, NEGLIGENCE OR OTHERWISE ARISING OUT OF THIS SOFTWARE OR THE DOCUMENTATION.

Module mod_fastcgi License

This FastCGI application library source and object code (the “Software”) and its documentation (the “Documentation”) are copyrighted by Open Market, Inc (“Open Market”). The following terms apply to all files associated with the Software and Documentation unless explicitly disclaimed in individual files.

Open Market permits you to use, copy, modify, distribute, and license this Software and the Documentation solely for the purpose of implementing the FastCGI specification defined by Open Market or derivative specifications publicly endorsed by Open Market and promulgated by an open standards organization and for no other purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions.

No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this Software and Documentation may be copyrighted by their authors and need not follow the licensing terms described here, but the modified Software and Documentation must be used for the sole purpose of implementing the FastCGI specification defined by Open Market or derivative specifications publicly endorsed by Open Market and promulgated by an open standards organization and for no other purpose. If modifications to this Software and Documentation have new licensing terms, the new terms must protect Open Market’s proprietary rights in the Software and Documentation to the same extent as these licensing terms and must be clearly indicated on the first page of each file where they apply.

Open Market shall retain all right, title and interest in and to the Software and Documentation, including without limitation all patent, copyright, trade secret and other proprietary rights.

OPEN MARKET MAKES NO EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE SOFTWARE OR THE DOCUMENTATION, INCLUDING WITHOUT LIMITATION ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL OPEN MARKET BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY DAMAGES ARISING FROM OR RELATING TO THIS SOFTWARE OR THE DOCUMENTATION, INCLUDING, WITHOUT LIMITATION, ANY INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES OR SIMILAR DAMAGES, INCLUDING LOST PROFITS OR LOST DATA, EVEN IF OPEN MARKET HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED “AS IS”. OPEN MARKET HAS NO LIABILITY IN CONTRACT, TORT, NEGLIGENCE OR OTHERWISE ARISING OUT OF THIS SOFTWARE OR THE DOCUMENTATION.

Index